TUGAS AKHIR – KI141502

GESTUR TANGAN SEBAGAI INTERAKSI DAN KONTROL DALAM REALITAS VIRTUAL MENGGUNAKAN GOOGLE CARDBOARD DAN LEAP MOTION

Tri Sutrisno Nusantara
NRP 5112100074

Dosen Pembimbing
Anny Yuniarti, S.Kom., M.Comp.Sc.
Ridho Rahman Hariadi, S.Kom., M.Sc.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2016

[Halaman ini sengaja dikosongkan]

FINAL PROJECT – KI141502

HAND GESTURE AS INTERACTION AND CONTROL IN VIRTUAL REALITY USING GOOGLE CARDBOARD AND LEAP MOTION

Tri Sutrisno Nusantara
NRP 5112100074

Advisor
Anny Yuniarti, S.Kom., M.Comp.Sc.
Ridho Rahman Hariadi, S.Kom., M.Sc.

DEPARTMENT OF INFORMATICS
Faculty of Information Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2016

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

Gestur Tangan Sebagai Interaksi dan Kontrol Dalam Realitas Virtual Menggunakan Google Cardboard dan Leap Motion

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Interaksi Grafis dan Seni
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh :
Tri Sutrisno Nusantara
NRP : 5112100074

Disetujui oleh Dosen Pembimbing Tugas Akhir :

ANNY YUNIARTI, S.Kom., M.Comp.Sc.
NIP: 198106222005012002 ..... (pembimbing 1)

RIDHO RAHMAN HARIADI, S.Kom., M.Sc.
NIP: 198702132014041001 ..... (pembimbing 2)

SURABAYA
JUNI 2016

[Halaman ini sengaja dikosongkan]

GESTUR TANGAN SEBAGAI INTERAKSI DAN KONTROL DALAM REALITAS VIRTUAL MENGGUNAKAN GOOGLE CARDBOARD DAN LEAP MOTION

Nama Mahasiswa : Tri Sutrisno Nusantara
NRP : 5112100074
Jurusan : Teknik Informatika FTIf-ITS
Dosen Pembimbing 1 : Anny Yuniarti, S.Kom., M.Comp.Sc.
Dosen Pembimbing 2 : Ridho Rahman Hariadi, S.Kom., M.Sc.

ABSTRAK

Realitas Virtual merupakan salah satu bidang teknologi yang sedang berkembang pesat. Banyak pengembang berlomba-lomba untuk mengembangkan aplikasi berbasis teknologi realitas virtual. Namun, terdapat keterbatasan dalam berinteraksi dalam lingkungan realitas virtual. Oleh karena itu, perlu dibuat sebuah interaksi dan kontrol berbasis gestur tangan pada lingkungan realitas virtual.

Memanfaatkan Leap Motion dalam menangkap gerakan tangan secara real-time, pengguna dapat berinteraksi secara langsung dengan lingkungan virtual. Google Cardboard digunakan sebagai kamera utama pengguna untuk merasakan pengalaman realitas virtual dengan smartphone.

Pengujian interaksi Leap Motion dalam realitas virtual dilakukan dengan pembuatan skenario perbandingan interaksi Leap Motion dengan interaksi yang sudah dikenal yaitu gamepad serta kuesioner setelah pengujian. Melalui pengujian tersebut dapat diperoleh kesimpulan bahwa pengguna lebih mudah menggunakan gamepad dibandingkan dengan Leap Motion ketika berinteraksi pada lingkungan virtual. Hal ini disebabkan karena pengguna lebih familiar dengan penggunaan gamepad dibandingkan dengan Leap Motion.

Kata kunci**: **Realitas Virtual, Interaksi Manusia dan Komputer, Leap Motion, Google Cardboard

Hand Gesture As Interaction And Control In Virtual Reality Using Google Cardboard And Leap Motion

Student Name	: Tri Sutrisno Nusantara
Student ID	: 5112100074
Major	: Teknik Informatika FTIf-ITS
Advisor 1	: Anny Yuniarti, S.Kom., M.Comp.Sc.
Advisor 2	: Ridho Rahman Hariadi, S.Kom, M.Sc.

ABSTRACT

Virtual reality is one of highly developed technology in recent day. Many developers are trying to develop virtual reality based technology. Unfortunately, there's a limitation in virtual reality to interact with virtual environment. So, there should be an interaction and control based on hand gesture to interact with virtual environment.

Using Leap Motion technology as real-time hand motion tracker, user can interact directly into virtual environment. Google Cardboard is used as the main camera in virtual reality.

Interaction testing is used to compare Leap Motion with gamepad as widely known interaction device. Interaction testing is done by performing a scenario and post-test quesionnaire. As the result, user is more familiar with gamepad, compared to Leap Motion as interaction in virtual environment.

Keyword : Virtual Reality, Human Computer Interaction, Leap Motion, Google Cardboard

[Halaman ini sengaja dikosongkan]

KATA PENGANTAR

Alhamdulillahirabbil'alamin, segala puji bagi Allah SWT atas segala karunia dan rahmat-Nya penulis dapat menyelesaikan Tugas Akhir yang berjudul :

"Gestur Tangan Sebagai Interaksi dan Kontrol Dalam Realitas Virtual Menggunakan Google Cardboard dan Leap Motion"

Harapan dari penulis semoga apa yang tertulis di dalam buku Tugas Akhir ini dapat bermanfaat bagi pengembangan ilmu pengetahuan saat ini, serta dapat memberikan kontribusi yang nyata.

Dalam pelaksanaan dan pembuatan Tugas Akhir ini tentunya sangat banyak bantuan yang penulis terima dari berbagai pihak, tanpa mengurangi rasa hormat penulis ingin mengucapkan terima kasih sebesar-besarnya kepada:

1. Allah SWT dan Nabi Muhammad SAW.
2. Keluarga yang senantiasa memberikan dukungan penuh baik secara moriil maupun materiil.
3. Ibu Anny Yuniarti selaku dosen pembimbing pertama yang telah bersedia meluangkan waktu untuk membimbing dan memotivasi penulis dalam mengerjakan tugas akhir.
4. Bapak Ridho Rahman Hariadi selaku dosen pembimbing kedua yang telah bersedia meluangkan waktu untuk memberikan saran dan perbaikan yang membangun dalam pembuatan tugas akhir.
5. Bapak/Ibu dosen, staf dan karyawan Jurusan Teknik Informatika ITS yang telah banyak memberikan dukungan, ilmu dan bimbingan yang tak ternilai harganya bagi penulis.
6. Teman-teman Departemen Media Informasi yang selalu memberikan kebahagiaan dan motivasi disaat penulis sedang buntu.

7. Rizaldi Tri Yanuar, Rizal Maulidan, dan Radhea Wicaksono Putra yang selalu membantu penulis dalam pengerjaan tugas akhir sejak awal.
8. Teman-teman penghuni laboraturium Interaksi, Grafika dan Seni yang selalu memberikan solusi dari masalah yang ada pada Tugas Akhir ini.
9. Putu Adhi Purwanto, Diagnosa Fenomena, dan Devanda Tamba yang selalu menghibur penulis setiap saat.
10. Serta pihak-pihak lain yang tidak dapat disebutkan disini yang telah banyak membantu penulis dalam penyusunan Tugas Akhir ini.

Penulis telah berusaha sebaik-baiknya dalam menyusun Tugas Akhir ini, namun penulis mohon maaf apabila terdapat kekurangan, kesalahan maupun kelalaian yang telah penulis lakukan. Kritik dan saran yang membangun dapat disampaikan sebagai bahan perbaikan selanjutnya.

Surabaya, 28 Juni 2016
Penulis

Tri Sutrisno Nusantara

DAFTAR ISI

[Halaman ini sengaja dikosongkan]

DAFTAR GAMBAR

[Halaman ini sengaja dikosongkan]

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

DAFTAR KODE SUMBER

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

Pada bab ini akan dipaparkan mengenai garis besar Tugas Akhir yang meliputi latar belakang, tujuan, rumusan dan batasan permasalahan, metodologi pembuatan Tugas Akhir, dan sistematika penulisan.

1.1. Latar Belakang

Realitas virtual menjadi salah satu hal yang sedang menjadi perbincangan di bidang teknologi informasi. Masyakarat makin familiar dengan realitas virtual karena semakin mudahnya mendapatkan pengalaman realitas virtual. Banyak pengembang menawarkan teknologi baru yang berhubungan dengan realitas virtual. Secara umum, realitas virtual saat ini memanfaatkan indera penglihatan manusia dengan *head-mounted display*. Pengguna realitas virtual merasakan pengalaman berada di dunia digital. Melalui realitas virtual, pengguna dapat melakukan berbagai hal di berbagai bidang, diantaranya di bidang kesehatan, hiburan, militer, manufaktur, dan masih banyak lagi.

Namun diantara pilihan realitas virtual yang ada, masih belum banyak pengembang yang mengeksplorasi kontrol pengguna. *Gamepad* menjadi pilihan utama dalam melakukan kontrol dalam realitas virtual. Pengguna memanipulasi gerak dan aksi dalam realitas virtual menggunakan kombinasi tombol dan arah. Hal tersebut tentunya menyulitkan pengguna karena dalam pemakaian realitas virtual, khususnya dengan *head-mounted display*, pengguna tidak bisa melihat tombol yang harus ditekan pada gamepad. Pengguna harus mengingat lokasi tombol sehingga aksi yang dilakukan dalam realitas virtual tidak maksimal. Selain itu, penggunaan gamepad tidak memberikan kesempatan kepada pengguna untuk berinteraksi secara langsung di dunia digital.

Untuk menyelesaikan masalah tersebut, perlu ada perangkat yang mampu menangkap gerakan manusia sebagai kontrol dalam realitas virtual. Salah satu perangkat yang dapat melakukan itu

adalah Leap Motion. Leap Motion merupakan sebuah produk yang dikembangkan oleh Michael Buckwald dan David Holz yang mampu menangkap gerakan manusia, khususnya bagian tangan. Leap Motion memanfaatkan teknologi infrared untuk menangkap gerakan tangan manusia dengan menampatkan tiga LED infrared tepat dibawah posisi tangan. Melalui Leap Motion ini diharapkan interaksi pengguna di dunia digital menjadi lebih immersive.

Dalam tugas akhir ini, akan dilakukan penggabungan antara teknologi realitas virtual head-mounted display dengan interaksi dan kontrol karakter menggunakan Leap Motion. Head-mounted display yang akan digunakan dalam tugas akhir ini adalah Google Cardboard. Google Cardboard memanfaatkan perangkat smartphone sebagai tampilan utama dalam realitas virtual. Leap Motion diharapkan dapat menjadi interaksi kontrol utama dalam realitas virtual menggunakan Google Cardboard. Pengguna diharapkan dapat melakukan navigasi karakter pada dunia realitas virtual.

1.2. Rumusan Permasalahan

Rumusan masalah yang diangkat dalam tugas akhir ini dapat dipaparkan sebagai berikut:

1. Bagaimana cara membuat aplikasi yang dapat mendemonstrasikan teknologi realitas virtual dengan interaksi tangan melalui Leap Motion?
2. Bagaimana gestur tangan yang sesuai untuk menggerakkan karakter dalam lingkungan realitas virtual dengan Leap Motion?
3. Bagaimana perbandingan interaksi dengan *gamepad* dan interaksi dengan Leap Motion ketika melakukan navigasi dalam realitas virtual?

1.3. Batasan Permasalahan

Beberapa batasan masalah yang terdapat dalam tugas akhir ini dapat dipaparkan sebagai berikut:

1. Aplikasi dijalankan pada *smartphone* dengan sistem operasi Android.
2. Menggunakan Google Cardboard sebagai *head-mounted display*
3. Lingkungan pengembangan yang digunakan menggunakan aplikasi Unity 3D dan bahasa pemrograman C#.
4. Menggunakan Leap Motion SDK untuk menangkap gerakan tangan.
5. Gerakan yang ditangkap sebatas gerakan tangan.
6. Navigasi yang dilakukan yaitu maju, mundur, kiri, kanan, dan berhenti
7. Jarak antara Leap Motion Controller dengan ruang *input* tidak lebih dari 30 cm.

1.4. Tujuan

Tujuan dari pembuatan tugas akhir ini antara lain :

1. Membuat aplikasi realitas virtual yang dapat menangkap gerakan tangan manusia sebagai interaksi dan kontrol.
2. Mengeksplorasi gestur tangan sebagai navigasi karakter dalam realitas virtual.
3. Mengetahui perbandingan efektivitas pemakaian kontrol menggunakan *gamepad* dengan Leap Motion dalam realitas virtual.

1.5. Manfaat

Manfaat yang diharapkan dari pengembangan aplikasi ini diantaranya:

1. Terciptanya interaksi baru pada realitas virtual sehingga mampu mendapatkan pengalaman yang lebih *immersive*.
2. Dapat menjadi referensi untuk penelitian lebih lanjut mengenai pengembangan leap motion.

1.6. Metodologi

Langkah-langkah yang ditempuh dalam pengerjaan Tugas Akhir ini yaitu:

1.6.1. Penyusunan proposal Tugas Akhir

Proposal tugas akhir ini berisi mengenai rencana pengembangan aplikasi yang memanfaatkan Leap Motion Controller untuk berinteraksi dengan lingkungan virtual. Pendahuluan ini terdiri atas hal yang menjadi latar belakang diajukannya usulan tugas akhir, rumusan masalah yang diangkat, batasan masalah untuk tugas akhir, tujuan dari pembuatan tugas akhir, dan manfaat dari hasil pembuatan tugas akhir. Selain itu dijabarkan pula tinjauan pustaka yang digunakan sebagai referensi pendukung pembuatan tugas akhir. Sub bab metodologi berisi penjelasan mengenai tahapan penyusunan tugas akhir mulai dari penyusunan proposal hingga penyusunan buku tugas akhir. Terdapat pula subbab jadwal kegiatan yang menjelaskan jadwal pengerjaan tugas akhir.

1.6.2. Studi literatur

Tahap studi literatur merupakan tahap pembelajaran dan pengumpulan informasi yang digunakan untuk mengimplementasikan Tugas Akhir. Tahap ini diawali dengan pengumpulan literatur, diskusi, eksplorasi teknologi dan pustaka, serta pemahaman dasar teori yang digunakan pada topik Tugas Akhir. Literatur-literatur yang dimaksud disebutkan sebagai berikut:

a. Realitas Virtual
b. Google Cardboard
c. Trinus VR
d. Leap Motion Controller
e. SDK Leap Motion
f. Unity 3D
g. *First Person Controller*

1.6.3. Analisis dan Perancangan Sistem

Pada tahap ini dilakukan analisis dan pendefinisian kebutuhan sistem untuk masalah yang sedang dihadapi. Selanjutnya, dilakukan perancangan sistem dengan beberapa tahap sebagai berikut:

a. Perancangan diagram kasus penggunaan.
b. Perancangan model 3D.
c. Perancangan antarmuka sistem.
d. Perancangan proses aplikasi.

1.6.4. Implementasi

Pada tahap ini dilakukan implementasi antar muka, implementasi model objek 3D dan implementasi proses yang telah didefinisikan pada bab Analisis dan Perancangan Sistem. Kemudian dilakukan integrasi aplikasi dengan perangkat Google Cardboard serta Leap Motion. Aplikasi ini dibangun pada *game engine* Unity.

1.6.5. Pengujian dan Evaluasi

Pada tahap ini dilakukan pengujian aplikasi kepada pengguna secara langsung. Pengujian aplikasi dilakukan untuk membandingkan efektivitas pemakaian Leap Motion dengan *gamepad* sebagai sarana interaksi dalam lingkungan virtual. Melalui skenario yang dibuat, kedua jenis interaksi dibandingkan dengan mencatat waktu penyelesaian dari pengguna. Tahapan-tahapan dari pengujian adalah sebagai berikut:

a. Pengujian fungsionalitas
   Sebelum melakukan pengujian interaksi, aplikasi dipastikan siap untuk dipakai melalui serangkaian percobaan berdasarkan metode blackbox testing dan whitebox testing.
b. Identifikasi pengguna

Sebelum menjalankan pengujian interaksi, pengguna diminta untuk memberikan informasi dasar seperti umur, jenis kelamin, pengalaman dalam memainkan realitas virtual, dan pengalaman dalam menggunakan interaksi *gamepad.* Lewat informasi dasar tersebut, pengujian yang akan dilakukan diharapkan sesuai dengan karakteristik pengguna pada pengujian kali ini.

c. Pengujian interaksi dengan skenario
   Pengguna diminta untuk menjalankan skenario pada dengan menggunakan interaksi yang diberikan. Pengguna akan diberikan 3 kali percobaan untuk setiap interaksi. Pada setiap percobaan, waktu yang ditempuh pengguna untuk mencapai target akan dihitung.
d. Kuesioner pengguna.
   Pengguna diminta untuk mengisi kuesioner yang terkait dengan interaksi yang dilakukan pada pengujian interaksi. Parameter dari kuesioner yang dilakukan mencakup faktor kualitas untuk teknik perjalanan yang *immersive* [1] diantaranya :
   i. Akurasi (ketepatan pengguna dalam mencapai target)
   ii. Kehadiran (pengguna merasa benar-benar berada pada lingkungan virtual)
   iii. *Ease of Learning* (kemampuan pengguna awam untuk menggunakan interaksi terkait)
   iv. *Ease of Use* (tingkat kesulitan pemakaian interaksi dari sudut pandang pengguna)
   v. Kenyamanan (ketika pengguna menjalankan simulasi, pengguna tidak merasa pusing, mual, atau sakit)

1.6.6. Penyusunan Buku Tugas Akhir

Pada tahap ini dilakukan pendokumentasian dan pelaporan dari seluruh konsep, dasar teori, implementasi, proses yang telah

dilakukan, dan hasil-hasil yang telah didapatkan selama pengerjaan Tugas Akhir.

1.7. Sistematika Penulisan

Buku Tugas Akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan Tugas Akhir ini. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku Tugas Akhir terdiri atas beberapa bagian seperti berikut ini.

Bab I Pendahuluan

Bab ini berisi latar belakang masalah, tujuan dan manfaat pembuatan Tugas Akhir, permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penyusunan Tugas Akhir.

Bab II Tinjauan Pustaka

Bab ini membahas beberapa teori penunjang yang berhubungan dengan pokok pembahasan dan mendasari pembuatan Tugas Akhir ini.

Bab III Analisis dan Perancangan Sistem

Bab ini membahas mengenai perancangan perangkat lunak. Perancangan perangkat lunak meliputi perancangan data, arsitektur, proses dan perancangan antarmuka pada aplikasi.

Bab IV Implementasi

Bab ini berisi implementasi dari perancangan perangkat lunak.

Bab V Pengujian dan Evaluasi

Bab ini membahas pengujian dari aplikasi yang dibuat dengan melihat keluaran yang dihasilkan oleh aplikasi dan evaluasi untuk mengetahui kemampuan aplikasi serta mengetahui penilaian aspek kegunaan (*usability*) dari perangkat lunak.

Bab VI Kesimpulan dan Saran

Bab ini berisi kesimpulan dari hasil pengujian yang dilakukan. Bab ini membahas saran-saran untuk pengembangan sistem lebih lanjut.

Daftar Pustaka

Merupakan daftar referensi yang digunakan untuk mengembangkan Tugas Akhir.

Lampiran

Merupakan bab tambahan yang berisi daftar istilah yang penting pada aplikasi ini.

BAB II
TINJAUAN PUSTAKA

Bab ini menjelaskan tentang tinjauan pustaka yang menjadi dasar pembuatan Tugas Akhir. Beberapa teori, pustaka, dan teknologi yang mendasari pengerjaan Tugas Akhir ini diantaranya meliputi definisi realitas virtual, Google Cardboard, Trinus VR, Leap Motion Controller, SDK Leap Motion, Unity 3D dan *First Person Controller*. Penjelasan secara khusus masing-masing tinjauan pustaka dapat dilihat pada masing-masing subbab berikut ini.

2.1. Realitas Virtual

Ada beberapa definisi mengenai realitas virtual dalam dunia grafika komputer. Salah satunya, realitas virtual merupakan grafis interaktif yang dimodelkan dalam tiga dimensi secara *real-time*, digabungkan dengan tampilan yang mampu memberikan pengguna pengalaman yang *immersive* di dunia dalam model serta manipulasi secara langsung [2]. Secara langsung, realitas virtual mampu memanipulasi panca indera manusia seolah-olah mereka berinteraksi langsung ke dunia digital. Realitas virtual menjadi paradigma baru dalam interaksi manusia dan komputer serta banyak peneliti yang mulai tertarik akan banyaknya manfaat di berbagai bidang.

Terdapat berbagai macam hasil *output* dari realitas virtual. Salah satu *output* yang memanfaatkan indera penglihatan dalam memanipulasi ruang adalah *head-mounted display*. *Head-mounted display* merupakan sebuah alat yang terdiri dari dua layar dipasangkan persis di depan mata. Gambar yang dihasilkan oleh layar dihasilkan dari posisi dan orientasi dari pengguna, disesuaikan oleh sebuah *tracker* [2].

2.2. Google Cardboard

Google Cardboard merupakan salah satu hasil output dari realitas virtual berupa *head-mounted display* yang dikembangkan oleh Google. Pengguna dapat membuat sendiri *head-mounted display* melalui pola potong yang telah dibuat oleh pengembang. *Head-mounted display* dari Google Cardboard terdiri dari karton sebagai penyangga utama, dua kaca optik yang diposisikan dekat pada mata sebagai reflektor layar, serta dua lempeng magnet yang ditempatkan di samping *head-mounted display* sehingga pengguna bisa mengendalikan *menu* pada Google Cardboard. Model keseluruhan dari *head-mounted display* Google Cardboard dapat dilihat pada Gambar 2.1.

Gambar 2.1 Model Keseluruhan head-mounted display dari Google Cardboard

Google Cardboard memanfaatkan *smartphone* sebagai layar utama yang digunakan untuk menampilkan hasil *output*. *Smartphone* kemudian ditaruh pada sebuah *placeholder* di depan *head-mounted display*. Luas sudut pandang (*field of view)* yang diterima oleh pengguna ketika menggunakan Google Cardboard

yaitu 90 derajat [3]. Tampilan pada layar *smartphone* ketika menjalankan aplikasi Cardboard dapat dilihat pada Gambar 2.2

Gambar 2.2 Contoh Tampilan Cardboard pada smartphone

Selain berperan sebagai layar utama pada realitas virtual, *smartphone* juga berperan sebagai *tracker*. Google Cardboard memanfaatkan tiga sensor pada *smartphone*, yaitu sensor *gyroscope*, sensor magnetis, dan *Near Field Communication* (NFC). Sensor *gyroscope* berguna untuk mengambil posisi dan orientasi dari pengguna ketika memakai Google Cardboard. Sensor magnetis berguna untuk mendeteksi pergerakan magnet yang tertanam di samping *head-mounted display*. Ketika *smartphone* ditempatkan pada *placeholder*, *smartphone* akan mengenali NFC sebagai penanda Google Cardboard. Apabila pada *smartphone* tidak terdapat NFC, maka dapat digantikan dengan memotret kode QR yang terletak di samping *head-mounted display* untuk mengenali Google Cardboard.

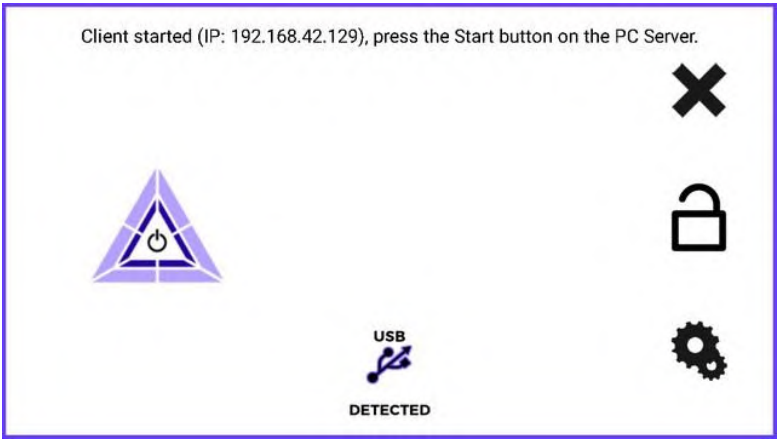
Gambar 2.3 Pengguna memakai Google Cardboard

Seperti yang terlihat pada Gambar 2.3, untuk memakai Google Cardboard, pengguna cukup memegang *head-mounted display* tepat di depan mata. Namun, apabila pengguna tidak ingin memegang *head-mounted display* ketika memakai Google Cardboard, pengguna bisa memasang sebuah *strap* yang dikaitkan pada kepala. Selain mudah untuk digunakan, bahan-bahan yang digunakan terjangkau sehingga pengguna tidak perlu mengeluarkan uang banyak demi merasakan pengalaman realitas virtual yang *immersive.*

Google Cardboard dapat digunakan pada sistem operasi Android dan iOS. Bagi pengembang aplikasi, Google menyediakan *Software Development Kit* (SDK) Cardboard. Melalui SDK Cardboard, Google menyediakan fitur-fitur yang membantu developer dalam mengonfigurasi semua sensor pada *smartphone* dan mengatasi distorsi yang dihasilkan oleh perbedaan ukuran *smartphone*. SDK Cardboard dapat diintegrasikan dengan *game engine* Unity3D untuk menciptakan model 3 dimensi [4].

2.3. Trinus VR

Trinus VR merupakan sebuah aplikasi yang menghubungkan komputer dengan *smartphone* sehingga dapat memainkan *game* realitas virtual secara terjangkau dengan menggunakan Google Cardboard. Secara umum, Trinus VR memanfaatkan sensor dan layar pada *smartphone* kemudian dikirimkan ke komputer untuk dapat memainkan *game* realitas virtual [5]. komputer bertindak sebagai *server* dalam pengaplikasian Trinus VR. Terdapat dua cara ntuk mengirimkan data dari *smartphone* ke komputer. Pertama, pengguna bisa menyambungkan perangkat *smartphone* melalui *wifi* dengan sambungan yang sama dengan komputer. Kedua, pengguna bisa menyambungkan perangkat *smartphone* ke komputer melalui kabel USB. Tampilan Trinus VR pada *smartphone* dapat dilihat pada Gambar 2.4.



Gambar 2.4 Aplikasi Trinus VR pada *smartphone* Android

Trinus VR menyediakan *Software Development Kit* (SDK) supaya dapat dikembangkan pada *game engine* Unity. Melalui SDK ini, pengembang dapat mengeksplorasi *library* yang disediakan untuk mengembangkan sebuah *game* virtual reality. *Prefab* yang disediakan oleh Trinus VR SDK terdiri dari Trinus

Camera, Trinus Manager, dan Trinus UI. Trinus Camera digunakan sebagai kamera utama pada permainan dengan sudut pandang orang pertama. Trinus Camera bergerak sesuai dengan sensor yang dikirimkan dari *smartphone*. Trinus Manager merupakan penghubung *library* utama dari Trinus VR. Trinus UI merupakan tampilan antar muka yang digunakan pada aplikasi. Trinus UI membantu pengguna untuk memulai sambungan antara *smartphone* dengan komputer melalui beberapa langkah.

2.4. Leap Motion Controller

Leap Motion Controller merupakan suatu perangkat yang dikembangkan oleh Leap Motion, Inc yang dikembangkan pada tahun 2008 oleh David Holz Leap Motion Controller sendiri digunakan sebagai *input* dari komputer tanpa sentuh, bisa dikatakan bahwa Leap Motion ini merupakan pengganti mouse, karena mempunyai tujuan dan fungsi yang sama. Pada tahun 2010, untuk pertama kalinya Leap Motion Controller ini diperkenalkan kepada publik [6]. Perkembangan dari Leap Motion Controller dapat dilihat pada Gambar 2.5.

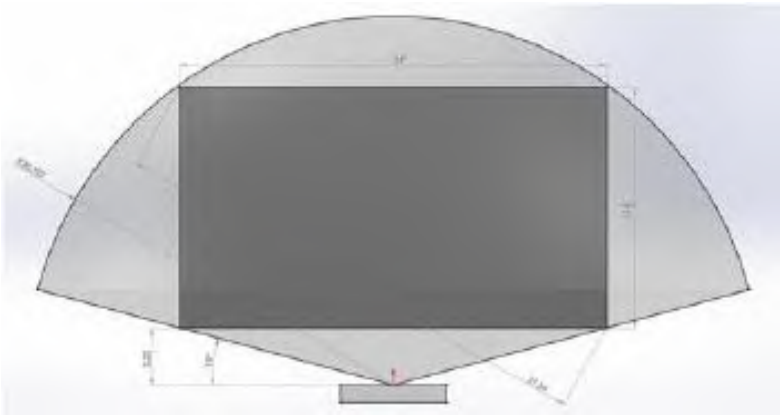
Gambar 2.5 Perkembangan Leap Motion Cntroller

Leap Motion seperti Gambar 2.6 sendiri merupakan penemuan penting dimana alat ini adalah alat sensor perangkat

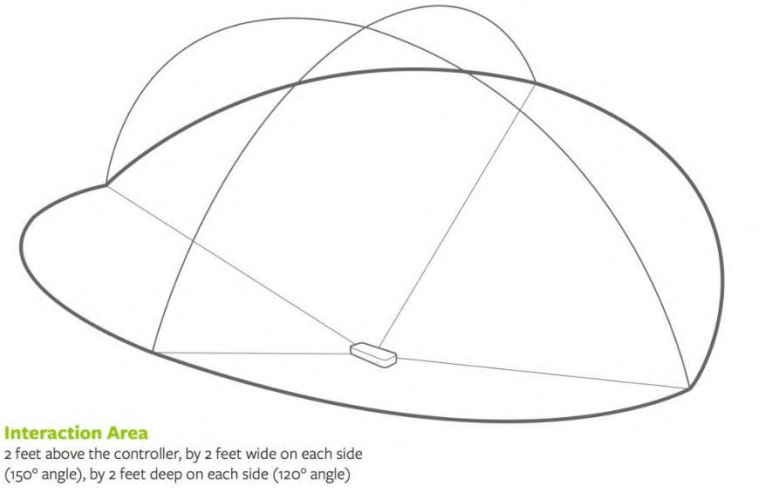
keras komputer yang mendukung gerakan tangan dan jari sebagai masukan, yang dapat disamakan fungsinya seperti *mouse*, namun tidak membutuhkan kontak langsung dengan tangan atau sentuhan sehingga diharapkan adanya sebuah pengalaman baru dalam dunia *virtual*

Gambar 2.6 Leap Motion Controller

Leap Motion dari sisi *hardware* sebenarnya cukup sederhana. Inti dari Leap Motion ini terletak pada pemanfaatan dua kamera *stereo* dan terdapat tiga lampu pemancar inframerah yang menyebar secara konvergen sehingga mampu untuk menjangkau area yang lebih luas. Jadi pada tahap ini inframerah akan menyebar untuk membentuk sebuah area seperti setengah lingkaran dengan jarak jangkauan maksimal 50 cm. Area jangkauan dari Leap Motion ini dapat dilihat pada Gambar 2.7 dan Gambar 2.8 [7].



Gambar 2.7 Area Jangkauan Inframerah dalam Dua Dimensi



Gambar 2.8 Area Jangkauan Inframerah dalam Tiga Dimensi

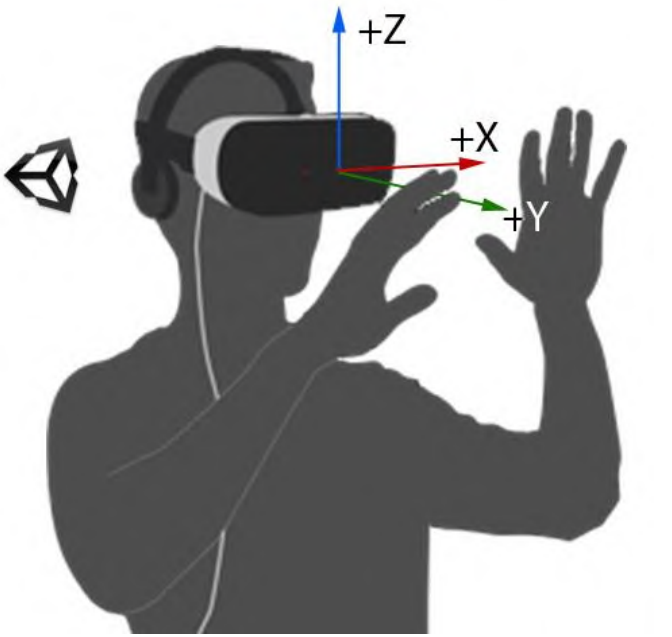
Leap Motion berjalan sebagai *service* atau *daemon* pada komputer. Untuk memproses gambar dan mengirim informasi,

Leap Motion membutuhkan sebuah S*oftware Development Kit* (SDK) [8].

2.5. SDK Leap Motion

Software Development Kit, atau yang biasa disebut SDK, adalah sebuah perangkat bantu pengembangan perangkat lunak yang digunakan untuk paket perangkat lunak tertentu, *framework* tertentu, perangkat keras tertentu, atau perangkat lain sejenisnya. Leap Motion SDK merupakan sekumpulan library yang berisikan tentang kebutuhan - kebutuhan sistem dari aplikasi yang digunakan untuk mengembangkan Leap Motion Controller ini. Biasanya terdapat beberapa fungsi yang dapat digunakan seperti pemodelan tangan dan jari - jari manusia. Leap Motion SDK ini dibuat untuk memudahkan developer dalam membangun aplikasi. Pengembang dapat mengunduhnya secara gratis dan tersedia dalam berbadai jenis bahasa pemrograman yang berbeda. Antara lain: Javascript, Unity / C#, C++, Java, Python, dan Objective-C. Juga mendukung berbagai macam sistem operasi yang berbeda yakni Windows, OSX, dan Linux.

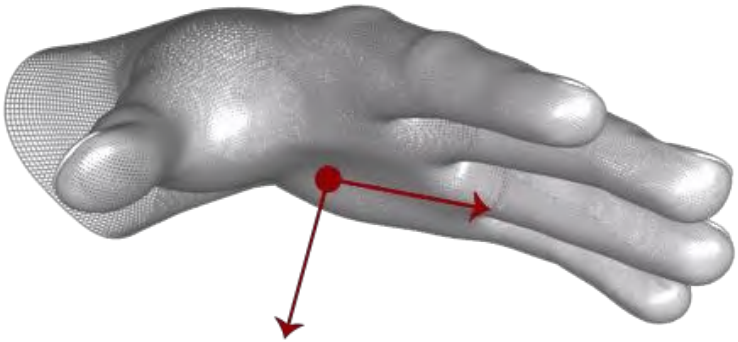
Salah satu fungsi pada Leap Motion SDK ini adalah mengolah data telapak tangan. Terdapat kelas utama yang mengengendalikan fungsi untuk mengolah data telapak tangan yaitu *HandModel*. Pada kelas *HandModel*, seluruh fungsi-fungsi telah mengikuti koordinat dari Unity. Koordinat dari Leap Motion secara dasar berbeda dengan koordinat Unity. Koordinat Unity menggunakan sistem konvensi tangan kiri, sedangkan Leap Motion menggunakan sistem konvensi tangan kanan. Untuk mengubah sistem koordinat dasar Leap Motion, terdapat fungsi *ToUnity()* yang mengembalikan koordinat dalam bentuk Vector3 ke sistem koordinat Unity [8]. Illustrasi penggunaan koordinat Unity pada Leap Motion dapat dilihat pada Gambar 2.9.



Gambar 2.9 Posisi Leap Motion pada Koordinat Unity

Pada kelas *HandModel* terdapat dua fungsi yang bermanfaat untuk mengetahui data posisi telapak tangan yaitu *GetPalmNormal()* dan *GetPalmDirection()*. *GetPalmNormal()* berfungsi mengembalikan projeksi vektor normal dari telapak tangan pengguna pada koordinat global [9]. Proyeksi vektor normal dari telapak tangan mengarah keluar dari permukaan dari telapak tangan. Proyeksi vektor normal dari telapak tangan berguna untuk mengetahui arah telapak tangan [10]. Posisi vektor normal dari telapak tangan dapat dilihat pada Gambar 2.10.

GetPalmDirection() berfungsi mengembalikan arah telapak tangan pada koordinat global. Fungsi ini berguna untuk menentukan posisi dari tangan ketika berada pada area jangkauan tangan dari Leap Motion.

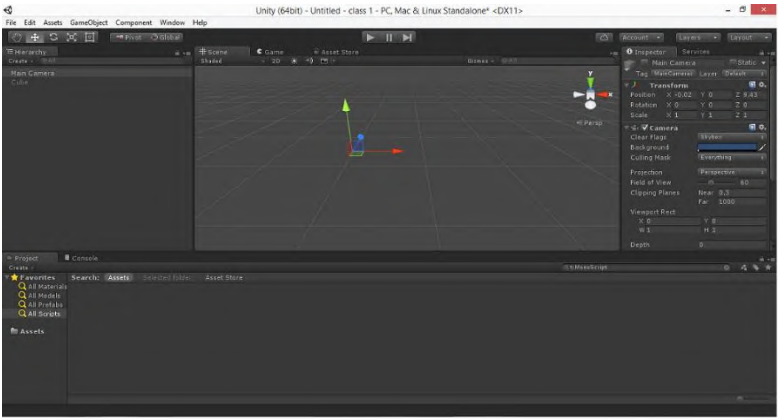


Gambar 2.10 Vektor Normal dari Telapak Tangan pada Leap Motion

2.6. Unity 3D

Unity atau Unity 3D adalah sebuah perangkat lunak yang berfungsi untuk membangun permainan atau aplikasi. Unity merupakan suatu *game development ecosystem* yang mampu digunakan untuk membuat permainan atau aplikasi dalam berbagai macam *platform* baik *console*, *desktop*, dan *mobile*. Bahasa pemrograman utama Unity adalah C# dengan IDE Mono Develop [11].

Unity 3D berbasis *cross-platform*, sehingga pengembang dapat membuat *game* yang dapat dimainkan pada perangkat komputer, ponsel pintar android, *web games* (memerlukan plugin unity *web player*), iPhone, PS3, dan bahkan X-BOX. Unity3D menyediakan *software free* dan *pro*, untuk versi gratis Unity menyediakan fitur pengembangan game berbasis windows, standlone mac dan web. Sedangkan untuk Unity *Pro* terdapat fitur yang lebih komplit dibandingkan dengan Unity Free seperti efek bayangan pada objek dan efek *water* yang lebih memukau. Tampilan antarmuka aplikasi ini dapat dilihat pada Gambar 2.11.



Gambar 2.11 Tampilan Antarmuka Unity 3D

2.7. First Person Controller

Salah satu genre permainan yang popular saat ini adalah *First Person Shooter* (FPS). FPS memanfaatkan sudut pandang orang pertama sebagai subjek utama dalam permainan. Dalam genre FPS, pengguna dapat secara bebas menggerakkan karakter dan memiliki keleluasaan dalam mengeksplorasi ruang layaknya dunia nyata.

Realitas Virtual memiliki keunggulan pada genre FPS karena pengguna seolah-olah terlibat secara nyata dalam lingkungan virtual. Namun, FPS pada realitas virtual terhambat oleh keleluasaan pengguna dalam mengeksporasi ruang. Oleh karena itu, dibutuhkan sarana interaksi yang *immersive* untuk mengontrol pergerakan karakter dalam lingkungan virtual.

Banyak penelitian yang dilakukan terkait interaksi yang dilakukan pada lingkungan virtual. Yoichi et al. [12] mengenalkan metode untuk mendeteksi gerakan tangan user dalam 3D dan mengenali gestur tangan secara *realtime* hanya dengan kamera, tanpa bantuan alat lain. Selain itu, terdapat penelitian yang dilakukan oleh Khundam [13] terkait penggunaan Leap Motion sebagai kontrol *first person* dengan memanfaatkan interaksi gestur

tangan pada realitas virtual. Dalam penelitian ini, Khundam menggabungkan Oculus Rift sebagai *head mount display* dan Leap Motion untuk mendeteksi gerakan tangan. Peneliti memanfaatkan posisi tangan dan vektor normal dari telapak tangan dalam mengenali gestur tangan pengguna. Terdapat 5 gestur tangan yang digunakan untuk pergerakan karakter dalam lingkungan virtual. Gerakan yang dilakukan mempertimbangkan sumbu horizontal pada koordinat Unity yaitu sumbu x dan z.

Supaya Leap Motion dapat mengenali gerakan pada sumbu z, Leap Motion perlu mengetahui vektor normal dari tangan. Vektor normal dapat mendefinisikan arah telapak tangan dari pengguna. Jadi, pengguna dapat memutar arah telapak tangan, seakan-akan pengguna dapat mengendalikan gerakan karakter. Ketika pengguna ingin menggerakkan karakter maju, pengguna diminta untuk menghadapkan telapak tangan keluar dari arah pandangan mata. Ketika pengguna ingin menggerakkan karakter mundur, pengguna diminta untuk menghadapkan telapak tangan menuju kearah pandangan mata.

Supaya Leap Motion dapat mengenali gerakan pada sumbu x, Leap Motion perlu mengetahui posisi tangan pengguna. Lewat posisi tangan dan pembentukan sudut pergelangan tangan dari posisi normal, aplikasi dapat mengenali gestur untuk pergerakan karakter ke kiri dan kanan. Ketika pengguna ingin menggerakkan karakter ke kiri, pengguna diminta untuk memutar pergelangan tangan ke kiri. Ketika pengguna ingin menggerakkan karakter ke kanan, pengguna diminta untuk memutar pergelangan tangan ke kanan.

Ketika pengguna ingin menghentikan gerakan karakter, perlu adanya perbedaan keadaan saat bergerak dan berhenti. Selama menggerakkan karakter, telapak tangan pengguna selalu membuka. Leap Motion perlu mengenali keadaan tangan saat membuka atau tidak. Oleh karena itu, ketika pengguna ingin menghentikan gerakan karakter, pengguna mengepalkan tangan. Untuk mengenali pengguna ketika akan berhenti, peneliti

memanfaatkan fungsi dalam SDK Leap Motion yaitu *pinchStrength()*.

BAB III
ANALISIS DAN PERANCANGAN SISTEM

Bab ini menjelaskan tentang analisis dan perancangan aplikasi dalam Google Cardboard yang memanfaatkan teknologi Leap Motion Controller untuk interaksi dan kontrol. Pembahasan yang akan dilakukan meliputi analisis fitur yang dibutuhkan dan perancangan perangkat lunak.

3.1. Analisis Perangkat Lunak

Selama ini, pengguna Google Cardboard tidak dapat menggerakan karakter dengan bebas. Pengguna hanya diberikan magnet untuk berinteraksi dalam dunia realitas virtual. Padahal, dengan pengalaman yang diberikan oleh Google Cardboard, seharusnya dapat diimbangi dengan interaksi yang memadai. Oleh karena itu, perlu ada perangkat yang membantu pengguna Google Cardboard supaya dapat berinteraksi secara leluasa.

Aplikasi ini dibangun dengan tujuan untuk mendemonstrasikan penggunaan Leap Motion sebagai interaksi dan kontrol dalam Google Cardboard. Aplikasi ini menampilkan model 3D sebuah rumah untuk dapat ditelusuri oleh pengguna dalam Google Cardboard.

Pembuatan aplikasi dilakukan dalam *game engine* Unity. Cardboard SDK digunakan dalam Unity supaya aplikasi dapat diterapkan dalam Google Cardboard. Selain itu, penggunaan SDK Leap Motion juga digunakan untuk menerima data tangan yang nantinya akan diolah dalam Unity.

3.1.1. Deskripsi Umum Perangkat Lunak

Tugas akhir ini memanfaatkan teknologi Leap Motion untuk membantu pengguna Google Cardboard dalam berinteraksi dalam dunia realitas virtual secara leluasa. Pengguna dapat memanfaatkan gestur tangan untuk menggerakkan karakter dalam Google Cardboard. Penerapan teknologi Leap Motion untuk menggerakan

karakter didemonstrasikan melalui sebuah aplikasi yang dapat menelusuri sebuah rumah. Aplikasi dibangun dalam *game engine* Unity dengan bantuan Cardboard SDK dan Leap Motion SDK. *First Person Camera* menjadi tampilan utama yang dapat dilihat oleh pengguna sehingga pengguna seolah-olah berada pada aplikasi ini.

Terdapat 5 macam gestur tangan yang dimanfaatkan untuk menggerakkan karakter dengan bantuan Leap Motion. Setiap gestur tangan yang ditunjukkan pada Leap Motion merepresentasikan arah karakter bergerak, yaitu maju, mundur, geser ke kiri, geser ke kanan dan berhenti. Apabila tidak terdeteksi tangan maka karakter akan berhenti. Apabila pengguna ingin berjalan menuju arah tertentu, pengguna dapat memutar posisi badan ke arah yang diinginkan sehingga Google Cardboard dapat mendeteksi perubahan posisi karakter.

Model 3D yang digunakan dalam aplikasi ini adalah sebuah rumah yang dapat dijelajahi oleh pengguna. Harapannya, aplikasi ini dapat menjadi contoh sebuah *virtual tour* yang memanfaatkan teknologi Leap Motion untuk menggerakkan karakter.

Untuk mengetahui efektivitas dan kemudahan pengguna dalam menggunakan Leap Motion, sebuah skenario diterapkan dalam aplikasi. Tujuan pembuatan skenario adalah untuk membandingkan efektivitas pemakaian Leap Motion dengan *gamepad* dalam menggerakkan karakter. Skenario dibangun dalam lingkungan rumah dan pengguna diminta untuk melewati beberapa *checkpoint* sesuai dengan urutan. Terdapat 12 *checkpoint* yang harus dilewati pengguna. Selama proses pengambilan *checkpoint*, terdapat pencatatan waktu pengguna ketika mulai menggerakkan karakter hingga selesai melewati semua *checkpoint.*

Terdapat beberapa tahapan agar pengguna dapat menggunakan aplikasi ini. Pertama, pengguna diminta untuk memilih interaksi yang akan dipakai dalam menjalankan karakter. Pilihan yang diberikan adalah *gamepad* dan Leap Motion. Pengguna juga harus memastikan interaksi yang dipilih telah tersambung pada perangkat. Kedua, pengguna diminta untuk

mengetahui letak *checkpoint* yang harus dilewati. Terakhir, pengguna menggerakkan karakter hingga semua *checkpoint* berhasil dilewati.

3.1.2. Spesifikasi Kebutuhan Fungsional Perangkat Lunak

Pada sistem ini, terdapat beberapa kebutuhan fungsional yang mendukung untuk jalannya aplikasi. Fungsi yang terdapat dalam aplikasi ini adalah sebagai berikut:

a. Menggerakkan karakter dengan *gamepad*
   Aplikasi ini harus mampu menggerakkan karakter yang ada dalam *scene* dengan menggunakan *gamepad*.
b. Menggerakkan karakter dengan Leap Motion
   Aplikasi ini harus mampu menggerakkan karakter yang ada dalam *scene* dengan memanfaatkan gestur tangan yang nantinya ditangkap oleh Leap Motion. Setiap gestur tangan yang ditangkap merepresentasikan arah gerak karakter.
c. Menampilkan *checkpoint*
   Aplikasi ini harus mampu menampilkan sejumlah *checkpoint* yang harus dilewati oleh pengguna secara berurutan.
d. Mencatat waktu tempuh
   Aplikasi ini harus mampu mencatat waktu yang ditempuh oleh pengguna ketika mulai permainan hingga seluruh *checkpoint* telah dilewati.

3.1.3. Identifikasi Pengguna

Aplikasi ini hanya memiliki satu aktor, yaitu orang yang menggunakan aplikasi dengan memanfaatkan Leap Motion sebagai interaksi dan kontrol.

3.2. Perancangan Perangkat Lunak

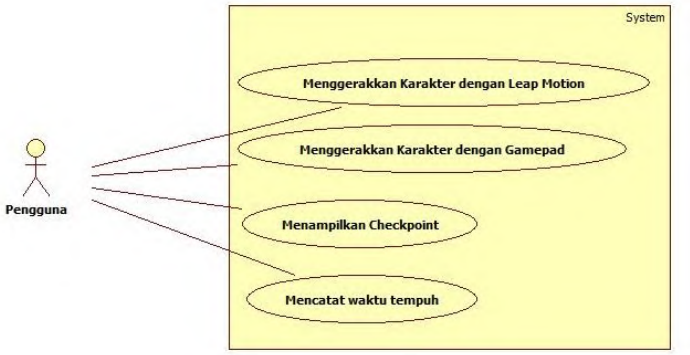
Subbab ini membahas bagaimana rancangan dari aplikasi tugas akhir ini. Meliputi: Model Kasus Penggunaan, Definisi Aktor, Definisi Kasus Penggunaan, Arsitektur Umum Sistem, Rancangan Antarmuka Aplikasi, dan Rancangan Proses Aplikasi.

3.2.1. Model Kasus Penggunaan

Berdasarkan analisis spesifikasi kebutuhan fungsional dan analisis aktor dari sistem dibuat kasus penggunaan sistem. Kasus-kasus penggunaan dalam sistem ini akan dijelaskan secara rinci pada subbab ini. Kasus penggunaan digambarkan dalam sebuah diagram kasus penggunaan. Diagram kasus penggunaan dapat dilihat pada Tabel 3.1 dan Gambar 3.1.

Tabel 3.1 Daftar Kasus penggunaan

Kode Kasus Penggunaan	Nama
UC-001	Menggerakkan karakter dengan Leap Motion
UC-002	Menggerakkan karakter dengan *gamepad*
UC-003	Menampilkan *checkpoint*
UC-004	Mencatat waktu tempuh



Gambar 3.1 Diagram Kasus Penggunaan

3.2.2. Definisi Kasus Penggunaan

Detail mengenai kasus penggunaan tersebut dapat dilihat pada subbab berikut ini.

3.2.2.1. Menggerakkan Karakter dengan Leap Motion

Spesifikasi kasus penggunaan menggerakkan arakter dengan Leap Motion dapat dilihat pada Tabel 3.2.

Tabel 3.2 Spesifikasi Kasus Menggerakkan Karakter dengan Leap Motion

Nama	Menggerakkan Karakter dengan Leap Motion
Kode	UC-001
Deskripsi	Pengguna dapat menggerakkan karakter dengan Leap Motion Controller
Aktor	Pengguna
Kondisi Awal	Pengguna sudah menyambungkan Leap Motion dengan aplikasi. Menu utama sudah dijalankan.
Aliran: **- Kejadian Normal**	1. Pengguna menunjukkan tangan kanan di depan Leap Motion dengan gestur telapak tangan menghadap depan A1.1. Pengguna menunjukkan tangan kanan di depan Leap Motion dengan gestur telapak tangan menghadap belakang A1.2. Pengguna menunjukkan tangan kanan di depan Leap Motion dengan gestur telapak tangan menghadap belakang dan miring ke kanan A1.3. Pengguna menunjukkan tangan kanan di depan Leap Motion dengan gestur telapak tangan

	menghadap belakang dan miring ke kiri A1.4. Pengguna menunjukkan tangan kanan di depan Leap Motion dengan gestur mengepal A1.5. Pengguna tidak menunjukkan tangan di depan Leap Motion 2. Sistem akan mengambil data tangan dari pengguna, menampilkan tangan virtual sesuai dengan gestur tangan pengguna dan mengubah posisi karakter menjadi ke depan 3. Pengguna menggerakkan Google Cardboard ke arah tertentu 4. Sistem akan memutar kamera sesuai dengan arah gerakan Google Cardboard 5. Kembali ke alur kejadian normal nomor 1
-Kejadian Alternatif	A1.1 Pengguna menunjukkan tangan kanan di depan Leap Motion dengan gestur telapak tangan menghadap belakang 1. Sistem akan mengambil data tangan dari pengguna, menampilkan tangan virtual sesuai dengan gestur tangan pengguna dan posisi karakter menjadi mundur 2. Kembali ke alur normal nomor 3 A1.2. Pengguna menunjukkan tangan kanan di depan Leap Motion dengan gestur telapak tangan menghadap belakang dan miring ke kanan 1. Sistem akan mengambil data tangan dari pengguna, menampilkan tangan virtual sesuai dengan gestur

	tangan pengguna dan menggerakkan karakter ke kanan 2. Kembali ke alur normal nomor 3 A1.3. Pengguna menunjukkan tangan kanan di depan Leap Motion dengan gestur telapak tangan menghadap belakang dan miring ke kiri 1. Sistem akan mengambil data tangan dari pengguna, menampilkan tangan virtual sesuai dengan gestur tangan pengguna dan menggerakkan karakter ke kiri 2. Kembali ke alur normal nomor 3 A1.4. Pengguna menunjukkan tangan kanan di depan Leap Motion dengan gestur mengepal 1. Sistem akan mengambil data tangan dari pengguna, menampilkan tangan virtual sesuai dengan gestur tangan pengguna dan berhenti menggerakkan karakter 2. Kembali ke alur normal nomor 3 A1.5. Pengguna tidak menunjukkan tangan di depan Leap Motion 1. Sistem akan menunjukkan bahwa tidak ada tangan yang ditampilkan dan berhenti menggerakkan karakter 2. Kembali ke alur normal nomor 3
Kondisi Akhir	Aplikasi akan menggerakan karakter sesuai dengan gestur yang dilakukan pengguna

3.2.2.2. Menggerakkan Karakter dengan *Gamepad*

Spesifikasi kasus penggunaan menggerakkan karakter dengan *gamepad* dapat dilihat pada Tabel 3.3.

Tabel 3.3 Spesifikasi Kasus Menggerakkan Karakter dengan *Gamepad*

Nama	Menggerakkan Karakter dengan *Gamepad*
Kode	UC-002
Deskripsi	Pengguna dapat menggerakkan karakter dengan *gamepad*
Aktor	Pengguna
Kondisi Awal	Pengguna sudah menyambungkan *gamepad* dengan aplikasi. Menu utama sudah dijalankan.
Aliran: **- Kejadian Normal**	1. Pengguna menggerakkan *analog* kiri ke arah atas pada *gamepad* A.1.1. Pengguna menggerakkan *analog* kiri ke arah kanan pada *gamepad* A.1.2. Pengguna menggerakkan *analog* kiri ke arah kiri pada *gamepad* A.1.3. Pengguna menggerakkan *analog* kiri ke arah bawah pada *gamepad* 2. Sistem akan menerima masukan dari *gamepad* dan menggerakkan karakter ke depan 3. Pengguna menggerakkan Google Cardboard ke arah tertentu 4. Sistem akan memutar kamera sesuai dengan arah gerakan Google Cardboard 5. Kembali ke alur kejadian normal nomor 1
-Kejadian Alternatif	A.1.1. Pengguna menggerakkan *analog* kiri ke arah kanan pada *gamepad*

	1. Sistem akan menerima masukan dari *gamepad* dan menggerakkan karakter ke kanan 2. Kembali ke alur normal nomor 3 A.1.2. Pengguna menggerakkan *analog* kiri ke arah kiri pada *gamepad* 1. Sistem akan menerima masukan dari *gamepad* dan menggerakkan karakter ke kiri 2. Kembali ke alur normal nomor 3 A.1.3. Pengguna menggerakkan *analog* kiri ke arah bawah pada *gamepad* 1. Sistem akan menerima masukan dari *gamepad* dan posisi karakter menjadi mundur 2. Kembali ke alur normal nomor 3
Kondisi Akhir	Aplikasi menggerakkan karakter sesuai dengan masukan yang diterima dari *gamepad*

3.2.2.3. Menampilkan *Checkpoint*

Spesifikasi kasus penggunaan Menampilkan *checkpoint* dapat dilihat pada Tabel 3.4.

Tabel 3.4 Spesifikasi Kasus Penggunaan Menampilkan *Checkpoint*

Nama	Menampilkan *Checkpoint*
Kode	UC-003
Deskripsi	Pengguna dapat melihat *checkpoint* dan melewati semua checkpoint yang ada secara berurutan
Aktor	Pengguna
Kondisi Awal	Salah satu masukan telah terhubung dan menu utama sudah dijalankan

Aliran: **- Kejadian Normal**	1. Pengguna memulai permaianan 2. Sistem menampilkan *checkpoint* dan menunjukkan jumlah *checkpoint* yang harus dilewati 3. Pengguna melewati *checkpoint* sesuai urutan A.3.1. Pengguna melewati *checkpoint* tidak sesuai urutan A.3.2. Pengguna telah melewati semua *checkpoint* yang disediakan 4. Sistem menandai bahwa *checkpoint* telah dilewati dan sistem menampilkan sisa *checkpoint* yang harus dilewati 5. Kembali ke alur normal nomor 3
-Kejadian Alternatif	A.3.1. Pengguna melewati *checkpoint* tidak sesuai urutan 1. Sistem akan memberitahu pengguna bahwa *checkpoint* yang dilewati tidak sesuai dengan urutan 2. Kembali ke alur normal nomor 2 A.3.2. Pengguna telah melewati semua *checkpoint* yang disediakan 1. Sistem akan menghentikan aktivitas pergerakan karakter dan menampilkan hasil akhir permainan
Kondisi Akhir	Pengguna berhasil melewati seluruh *checkpoint* yang diberikan

3.2.2.4. Mencatat Waktu Tempuh

Spesifikasi kasus penggunaan Mencatat Waktu Tempuh dapat dilihat pada Tabel 3.5.

Tabel 3.5 Spesifikasi Kasus Penggunaan Merekam Suara

Nama	Mencatat Waktu Tempuh

Kode	UC-004
Deskripsi	Pengguna dapat mengetahui waktu tempuh yang diperoleh dari memulai skenario hingga selesai melewati semua *checkpoint*
Aktor	Pengguna
Kondisi Awal	Menu utama sudah dijalankan
Aliran: **- Kejadian Normal**	1. Pengguna menjalankan skenario 2. Sistem mulai menjalankan waktu 3. Pengguna berhasil melewati semua *checkpoint* 4. Sistem berhenti menjalankan waktu dan mencatat waktu tempuh selama skenario berlangsung
-Kejadian Alternatif	-
Kondisi Akhir	Aplikasi akan mengeluarkan hasil waktu tempuh

3.2.3. Definisi Aktor

Aktor yang terdapat dalam aplikasi terlihat pada Tabel 3.6.

Tabel 3.6 Deskripsi Pengguna

No	Nama	Deskripsi
1	Pengguna	- Merupakan aktor yang bertugas menjalankan skenario yang telah disediakan dan memiliki hak untuk menggerakkan karakter dalam skenario - Aktor harus paham terlebih dahulu cara menggerakkan karakter dengan interaksi yang disediakan.

3.2.4. Perancangan Model 3D

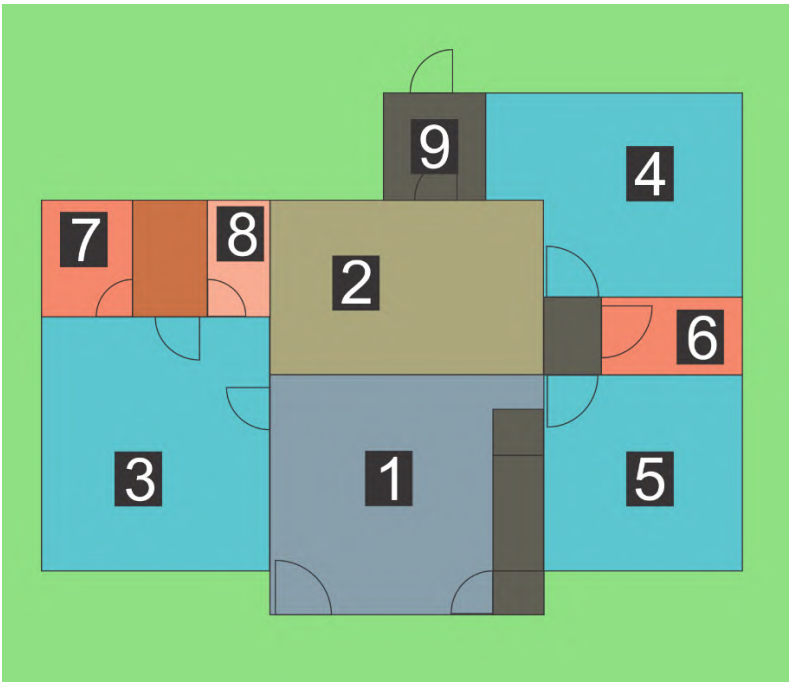
Model 3D yang dipakai adalah sebuah rumah. Model 3D tersebut digunakan sebagai lingkungan utama dalam skenario yang akan dibuat. Model 3D rumah dapat diunduh melalui situs http://www.blendswap.com/. Nantinya model 3D akan diimpor pada Unity dalam format fbx. Detil rancangan model ditampilkan pada sub bab berikutnya.

3.2.4.1. Perancangan Model 3D Rumah

Rancangan model 3D rumah tampak pada Gambar 3.2. Perancangan model 3D rumah terdiri dari 7 bagian yag dapat diakses diantaranya:

1. Ruang Tamu
2. Dapur
3. Kamar Tidur 1
4. Kamar Tidur 2
5. Kamar Tidur 3
6. Kamar Mandi 1
7. Kamar Mandi 2
8. Ruang Baju
9. Ruang Belakang

Berdasarkan rancangan model 3D tersebut, *checkpoint* akan ditaruh pada tiap ruangan yang tersedia. Selain itu, model 3D akan diberikan *collider* secara keseluruhan sehingga karakter yang terdapat dalam *scene* akan menabrak dinding dari rumah pada rancangan model 3D ketika karakter diarahkan pada dinding rumah. Apabila tidak diberikan *collider,* karakter akan menembus dinding rumah.

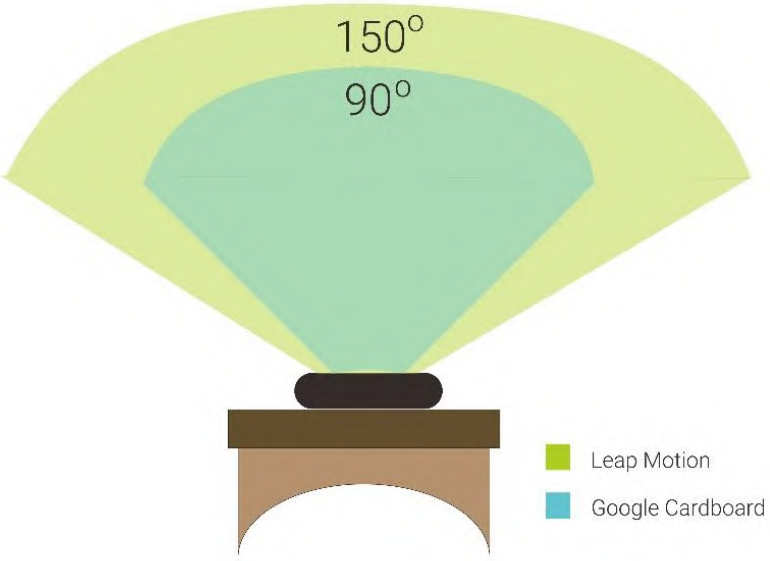


Gambar 3.2 Perancangan Model 3D Rumah Tampak Atas

3.2.5. Arsitektur Umum Sistem

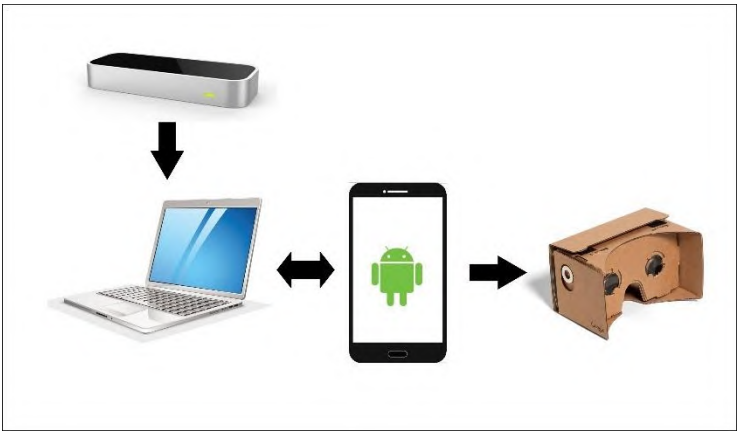
Arsitektur sistem pada aplikasi yang dibuat didukung oleh beberapa perangkat yaitu laptop, Leap Motion Controller, *gamepad*, *smartphone* dan Google Cardboard. Implementasi aplikasi dilakukan berbeda antara menggunakan Leap Motion dengan *gamepad.* Ketika menggunakan Leap Motion, aplikasi akan dijalankan pada laptop, kemudian data gambar akan diolah oleh Trinus VR untuk memproyeksikan gambar pada *smartphone*. Gambar yang ditampilkan oleh Trinus VR sudah diproses sehingga dapat digunakan pada Google Cardboard. Leap Motion akan diposisikan di depan Google Cardboard supaya jangkauan sensor pada Leap Motion mampu menangkap gerakan pengguna lebih leluasa [13]. Memanfaatkan *field of view* dari Leap Motion yang

luas serta posisi pada Google Cardboard memungkinkan jangkauan gerak tangan lebih luas, sehingga mengurangi kemungkinan Leap Motion tidak mampu menjangkau posisi tangan pengguna. Visualisasi posisi Leap Motion pada Google Cardboard beserta *field of view* dari masing-masing perangkat dapat dilihat pada Gambar 3.3.

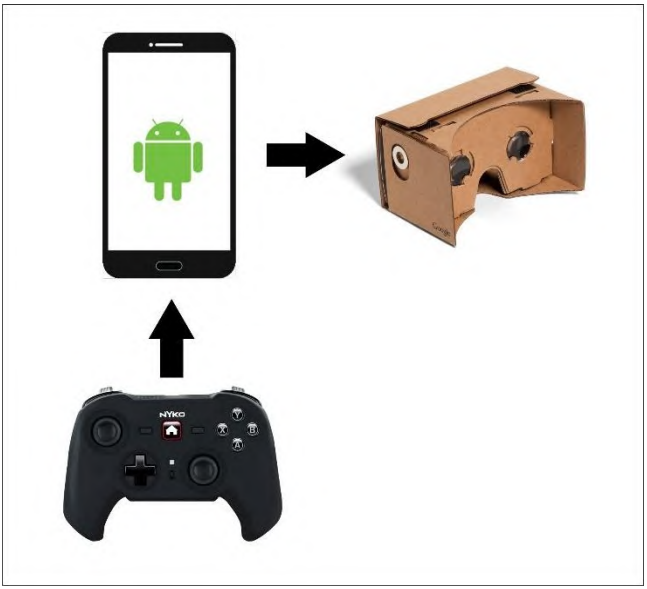


Gambar 3.3 Posisi perangkat Leap Motion pada Google Cardboard

Ketika menggunakan *gamepad*, aplikasi secara langsung dijalankan pada *smartphone*. Google Cardboard dipakai untuk membantu aplikasi pada *smartphone* menampilkan objek 3D secara *immersive*. Arsitektur secara umum dapat dilihat pada Gambar 3.4 dan Gambar 3.5.



Gambar 3.4 Rancangan Arsitektur Aplikasi Untuk Pergerakan Karakter dengan Leap Motion



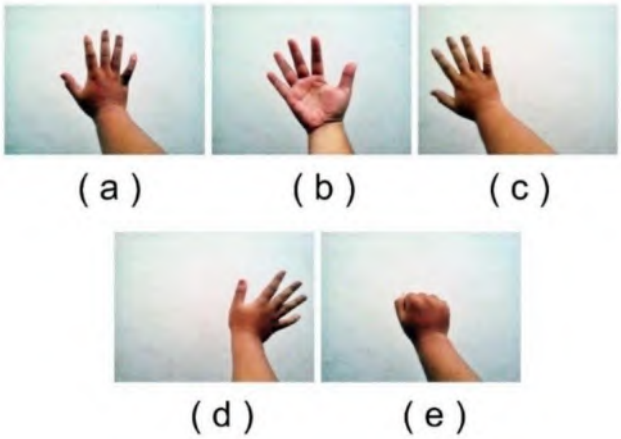
Gambar 3.5 Rancangan Arsitektur Aplikasi Untuk Pergerakan Karakter dengan Gamepad

3.2.6. Rancangan Gestur Tangan Untuk Pergerakan Karakter

Dalam menggerakkan karakter pada lingkungan virtual, Leap Motion perlu mengenali gerakan tangan tertentu. Definisi gestur tangan diperlukan agar lebih mudah dalam mengenali gerakan tangan yang merepresentasikan arah gerak karakter. Dalam penelitian sebelumnya, ketika merepresentasikan arah gerak karakter, terdapat 2 sumbu yang dipertimbangkan berdasarkan sistem koordinat Unity yaitu sumbu x dan sumbu z. Sumbu x digunakan untuk mengenali arah kiri dan kanan. Sumbu z digunakan untuk mengenali arah maju dan mundur. Oleh karena itu, terdapat 4 gestur tangan dalam pergerakan karakter yang akan digunakan. Supaya pengguna dapat lebih leluasa dalam menghentikan karakter, ditambahkan satu gestur tambahan untuk menghentikan gerakan karakter. Gestur untuk pergerakan maju, mundur, dan berhenti mengikuti penelitian yang dilakukan sebelumnya. Gestur pergerakan maju direpresentasikan dengan tangan yang membuka ke arah luar pandangan mata. Gestur pergerakan mundur direpresentasikan dengan tangan yang memuka menuju pandangan mata. Gestur menghentikan karakter direpresentasikan dengan tangan yang mengepal.

Terdapat perbedaan pada penelitian sebelumnya terkait pergerakan karakter ke kiri dan kanan. Supaya Leap Motion dapat mengenali gerakan pada sumbu x, Leap Motion perlu mengetahui arah tangan dari pengguna. Oleh karena itu, gestur tangan yang sesuai ketika pengguna ingin menggerakkan karakter pada arah gerak horizontal yaitu mencondongkan tangan secara horizontal dari posisi normal. Ketika pengguna ingin menggerakkan karakter ke kiri, pengguna diminta untuk menghadapkan telapak tangan keluar dari arah pandangan dan condongkan telapak tangan ke kiri. Ketika pengguna ingin menggerakkan karakter ke kanan, pengguna diminta untuk menghadapkan telapak tangan keluar dari arah pandangan dan condongkan telapak tangan ke kanan.

Illustrasi gestur tangan yang digunakan sebagai pergerakan karakter dapat dilihat pada Gambar 3.6.



Gambar 3.6 Gestur Tangan yang Merepresentasi Gerak Karakter (a) maju (b) mundur (c) kiri (d) kanan (e) berhenti

3.2.7. Rancangan Antarmuka Aplikasi

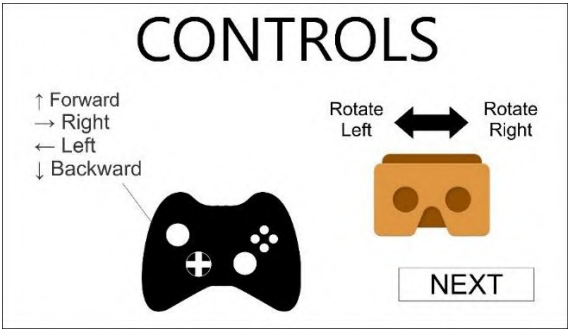
Rancangan antarmuka aplikasi diperlukan untuk memberikan gambaran umum kepada pengguna bagaimana sistem yang ada dalam aplikasi ini berinteraksi dengan pengguna. Selain itu, rancangan ini juga memberikan gambaran bagi pengguna apakah tampilan yang sudah disediakan oleh aplikasi mudah untuk dipahami dan digunakan, sehingga akan muncul kesan *user experience* yang baik dan mudah. Antar muka dari aplikasi ini secara umum menggunakan tampilan *first person*.

Terdapat menu utama yang mengantarkan pengguna untuk memulai permainan. Rancangan tampilan menu utama pertama pada permainan dapat dilihat pada Gambar 3.7. Selanjutnya pada menu utama kedua, pengguna dapat mempelajari kontrol dari permainan. Pada Gambar 3.8, rancangan tampilan menu tersebut digunakan untuk mengetahui cara menggerakkan karakter dengan *gamepad*. Pada Gambar 3.9, rancangan tampilan menu tersebut digunakan untuk mengetahui cara menggerakkan karakter dengan Leap Motion. Setelah menu utama kedua, terdapat menu utama

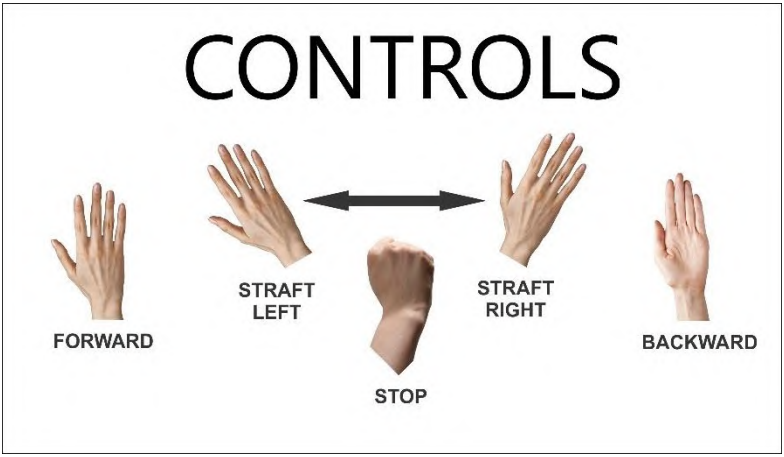
ketiga yang menampilkan peta berisi lokasi checkpoint serta petunjuk permainan. Rancangan menu utama ketiga dapat dilihat pada Gambar 3.10.

Setelah pengguna selesai melihat menu utama, pengguna dapat menggerakkan karakter dalam tampilan *first person* beserta indikator waktu dan sisa *checkpoint.* Rancangan tampilan *first person* dalam permainan dapat dilihat pada Gambar 3.11. Setelah pengguna berhasil memperoleh semua *checkpoint* sesuai dengan urutan yang diberikan, maka di akhir permainan akan muncul tampilan menu akhir. Menu ini berisikan waktu tempuh selama pengguna mengambil *checkpoint* hingga seluruh *checkpoint* terambil. Rancangan menu akhir dapat dilihat pada Gambar 3.12.

Gambar 3.7 Tampilan Menu Utama Saat Pertama Kali Masuk

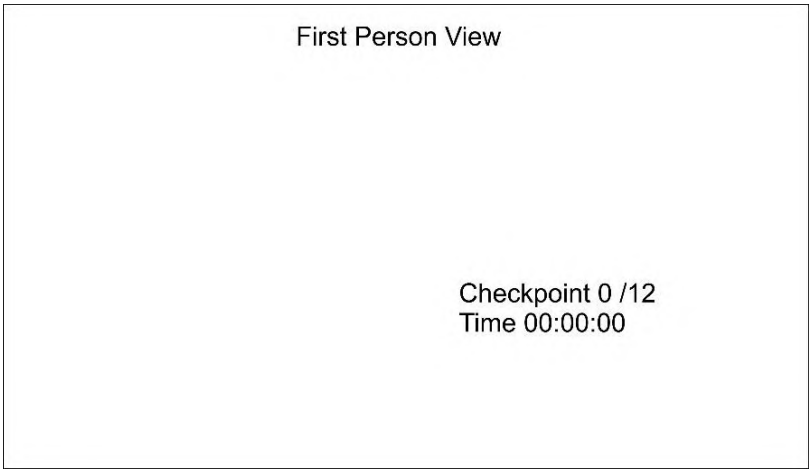


Gambar 3.8 Tampilan Menu Utama Untuk Memberitahu Pengguna Cara Menggerakkan Karakter dengan Gamepad



Gambar 3.9 Tampilan Menu Utama untuk Memberitahu Pengguna Cara Menggerakan Karakter dengan Leap Motion

Gambar 3.10 Tampilan Menu Utama Untuk Memberitahu Denah Rumah dan Lokasi Checkpoint



Gambar 3.11 Tampilan First Person Pengguna Ketika Masuk Dalam Skenario

Gambar 3.12 Tampilan Menu Setelah Pengguna Berhasil Melewati Semua Checkpoint

3.2.8. Perancangan Proses Aplikasi

Pada subbab ini akan dijelaskan mengenai rancangan proses yang dilakukan untuk mendukung fungsionalitas yang sudah

dirancang pada aplikasi. Rancangan ini diperlukan untuk memetakan proses yang ada mulai dari awal hingga akhir. Proses-proses terdiri dari proses melihat menu utama, proses melihat *checkpoint*, proses mencatat waktu tempuh, proses menggerakkan karakter dengan Leap Motion, dan proses menggerakkan karakter dengan *gamepad*.

3.2.8.1. Rancangan Proses Melihat Menu Utama

Proses ini merupakan tampilan pertama yang akan diperoleh pengguna ketika menjalankan aplikasi. Pengguna dapat melanjutkan aplikasi dengan menekan tombol "start". Selanjutnya, pengguna akan disajikan cara menggerakkan karakter pada permainan. Masing-masing interaksi memiliki cara menggerakkan karakter yang berbeda. Setelah pengguna menekan tombol "next", pengguna akan mendapatkan informasi mengenai cara bermain dan letak *checkpoint* yang harus dilewati selama skenario berjalan. Selanjutnya ketika pengguna siap untuk masuk ke dalam permainan, pengguna diminta menekan tombol "let's play". Pengguna diberi waktu untuk bersiap-siap sebelum menjalankan permainan.

3.2.8.2. Rancangan Proses Melihat *Checkpoint*

Proses ini merupakan proses utama yang harus dijalankan pengguna setelah melihat menu utama. Pengguna diharapkan untuk bersiap dalam menggerakkan karakter. Kemudian, sistem akan menampilkan seluruh *checkpoint* yang harus dilewati. Pengguna harus melewati *checkpoint* secara berurutan. Ketika pengguna berhasil melewati *checkpoint*, maka tampilan *checkpoint* akan hilang dan menambah daftar *checkpoint* yang telah dilewati. *Checkpoint* dibuat dengan menggunakan *box collider* pada Unity. Ketika karakter mengenai *collider checkpoint* maka proses *checkpoint* akan berjalan. Sisa *checkpoint* akan ditampilkan pada layar pemain. Letak dari *checkpoint* pada rumah dapat dilihat pada Gambar 3.13.

Gambar 3.13 Letak *checkpoint* pada rumah

3.2.8.3. Rancangan Proses Mencatat Waktu Tempuh

Proses ini merupakan proses yang berjalan bersamaan dengan proses melihat *checkpoint*. Ketika pengguna mulai masuk dalam skenario, sistem mulai menghitung waktu yang berjalan. Waktu tersebut akan ditampilkan pada layar pemain. Setelah pengguna berhasil melewati semua *checkpoint* dalam skenario, maka waktu akan berhenti dan akan ditampilkan pada menu akhir.

3.2.8.4. Rancangan Proses Menggerakkan Karakter dengan Leap Motion

Proses ini merupakan proses yang harus dijalankan pengguna untuk dapat melewati *checkpoint* dan menyelesaikan skenario. Leap Motion SDK dipakai untuk memperoleh gestur tangan pengguna. Pada proses ini juga diperlukan posisi kamera dari pemain yang dapat diambil dari kamera Trinus VR.

3.2.8.5. Rancangan Proses Menggerakkan Karakter dengan *Gamepad*

Proses ini merupakan proses yang dijalankan pengguna ketika melewati *checkpoint* dan menyelesaikan skenario dengan menggunakan *gamepad*. Pengguna memiliki hak untuk menggerakkan karakter dengan *analog* pada *gamepad.* Setiap arah dari *analog* merepresentasikan arah jalan dari karakter dalam lingkungan virtual. Apabila pengguna ingin memutar posisi karakter, maka pengguna diminta untuk memutar posisi Cardboard ke arah yang diinginkan. Proses menggerakkan karakter dengan *gamepad* akan diimplementasikan pada Unity dengan menggunakan *standard assets*.

3.2.9. Rancangan Kuesioner Interaksi

Parameter dari kuesioner mengacu pada faktor kualitas untuk teknik perjalanan yang *immersive.* Kuesioner dilakukan untuk masing-masing interaksi yang telah diujikan, yaitu *gamepad* dan Leap Motion. Kuesioner pengguna dapat dilihat pada Tabel 5.3.

Tabel 3.7 Kuesioner pengujian interaksi

No.	Parameter	Skala Nilai (1 - 6)
1.	[Akurasi] Saya merasa interaksi yang digunakan membantu mencapai target	
2.	[*Ease of Learning*] Saya merasa interaksi yang digunakan mudah dipelajari	
3.	[Kehadiran] Saya merasa berada di lokasi sesungguhnya ketika menggunakan interaksi ini	
4.	[*Ease of Use*] Saya merasa tidak kesulitan dalam menggunakan interaksi ini	
5.	[Kenyamanan] Saya merasa nyaman ketika menggunakan interaksi ini	

[Halaman ini sengaja dikosongkan]

BAB IV
IMPLEMENTASI

Bab ini akan menjelaskan tentang implementasi Tugas Akhir berdasarkan rancangan perangkat lunak. Proses implementasi mengacu pada rancangan perangkat yang telah dilakukan sebelumnya, namun juga dimungkinkan terjadinya perubahan-perubahan jika dirasa perlu. Implementasi dilakukan dalam bahasa C#.

4.1. Lingkungan Implementasi

Sub bab ini menjelaskan tentang lingkungan implementasi perangkat lunak yang dibangun. Lingkungan selama proses implementasi aplikasi adalah sebagai berikut:

Tabel 4.1 Spesifikasi Lingkungan Implementasi

Perangkat Keras	- Laptop Asus A43SD
	  o Prosesor Intel(R) Core(TM) i5-2450M CPU @ 2.50GHz
	  o RAM 8 GB
	- Leap motion Controller
	- Google Cardboard
	- *Gamepad*
	- *Smartphone* Mito Impact A10
	  o Prosesor Cortex A7 @1.3 GHz Quad-core
	  o RAM 1 GB

Perangkat Lunak	- Laptop Asus A43SD
	  o Sistem Operasi Microsoft Windows 10 Pro 64-bit
	  o Unity 5.3.4f1
	  o Leap Motion SDK
	  o Google Cardboard SDK
	- *Smartphone* Mito Impact A10
	  o *Sistem Operasi Android*

4.2. Implementasi Antar muka

Subbab ini akan menjelaskan tentang implementasi dari antar muka yang digunakan. Antarmuka aplikasi ini terdiri dari empat tampilan.

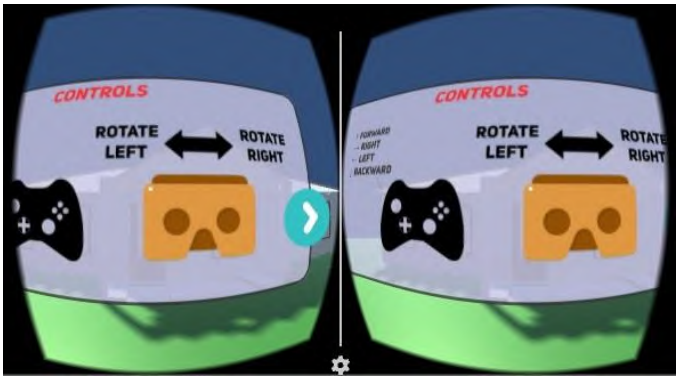
4.2.1. Impelementasi Antarmuka Menu Utama

Tampilan pertama merupakan menu utama. Pada tampilan ini terdapat satu tombol yaitu "*start*" yang berfungsi untuk memulai proses menjalankan skenario. Tampilan antarmuka pada menu utama dapat dilihat pada Gambar 4.1.

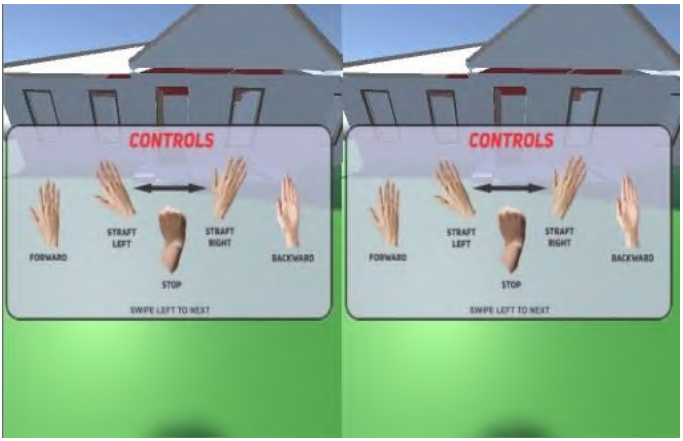
Gambar 4.1 Implementasi Antarmuka Menu Utama

4.2.2. Impelementasi Antarmuka Tata Cara Permainan

Tampilan kedua merupakan tampilan drum. Pada tampilan ini terdapat satu tombol, yaitu tombol "next". Tampilan ini juga menampilkan tata cara menjalankan skenario. Tampilan antarmuka tata cara permainan dapat dilihat pada Gambar 4.2 dan Gambar 4.3.



Gambar 4.2 Implementasi Antarmuka Tata Cara Permainan dengan Gamepad

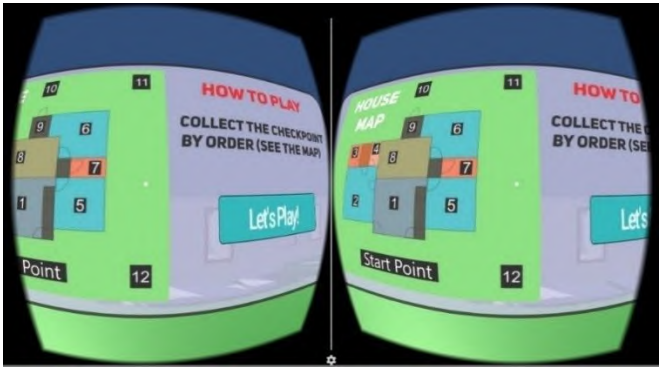


Gambar 4.3 Implementasi Antarmuka Tata Cara Permainan dengan Leap Motion

4.2.3. Impelementasi Antarmuka Petunjuk Peta

Tampilan ketiga merupakan tampilan petunjuk peta. Tampilan ini menunjukkan letak *checkpoint.* Terdapat satu tombol pada tampilan ini yaitu tombol “Let’s Play!” yang menandakan

pengguna siap untuk menjalankan skenario. Tampilan antarmuka petunjuk peta dapat dilihat pada Gambar 4.4.



Gambar 4.4 Implementasi Antarmuka Petunjuk Peta

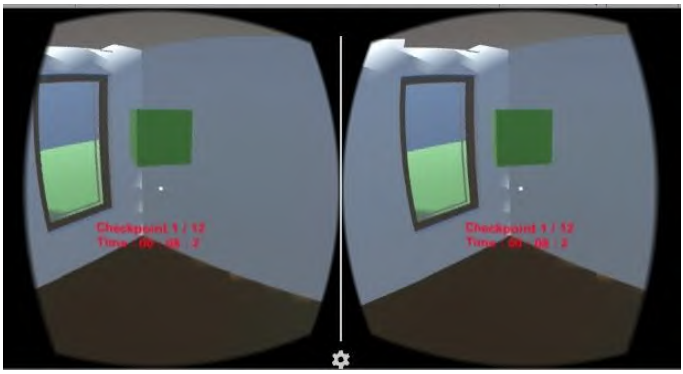
4.2.4. Implementasi Antarmuka Menu Akhir

Tampilan keempat merupakan tampilan menu akhir. Tampilan ini merupakan tampilan ketika pengguna telah berhasil melewati seluruh *checkpoint*. Waktu peneyelesaian pengguna juga ditampilkan pada menu ini. Tampilan antarmuka menu akhir dapat dilihat pada Gambar 4.5.

Gambar 4.5 Implementasi Antarmuka Menu Akhir

4.2.5. Implementasi Antarmuka Pemain

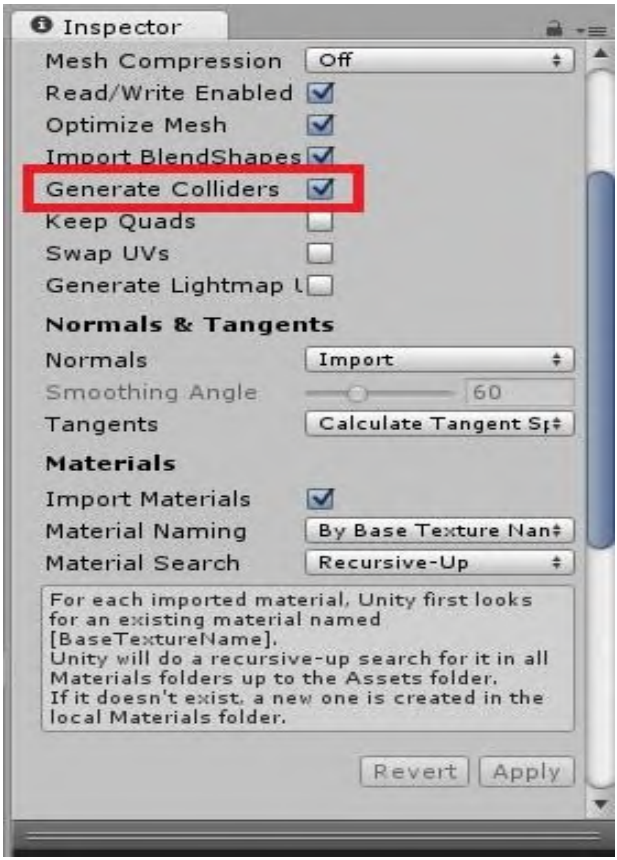
Tampilan kelima merupakan tampilan pemain. Tampilan ini merupakan tampilan ketika pengguna dapat menggerakkan karakter. Terdapat dua parameter yang ditunjukkan yaitu “Checkpoint” dan “Time”. “Checkpoint” berguna menampilkan sisa *checkpoint* yang harus dilewati pengguna. “Time” berguna menampilkan waktu yang telah dilewati selama menjalankan skenario. Tampilan antarmuka pemain dapat dilihat pada Gambar 4.6.



Gambar 4.6 Implementasi Antarmuka Pemain

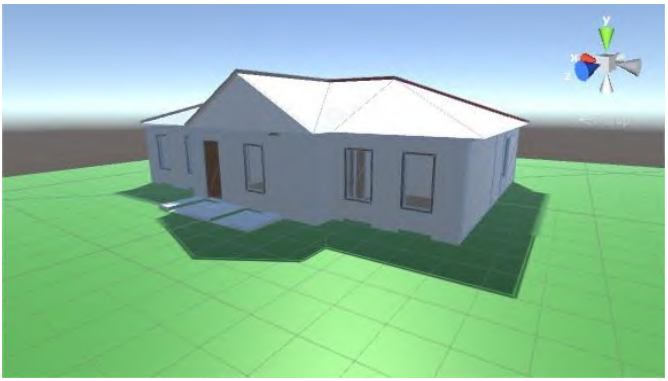
4.3. Implementasi Model 3D

Sub bab ini akan menjelaskan tentang implementasi dari model 3D yang telah dijelaskan pada Bab III. Implementasi model 3D diawali dengan mengimpor model 3D ke dalam Unity. Setelah diimpor ke dalam Unity, model 3D diberikan *collider*. Untuk mengimplementasikan *collider* pada model 3D secara otomatis, pilih *generate collider* pada *inspector* model 3D. Pilihan *generate collider* pada *inspector* model 3D dapat dilihat pada Gambar 4.7.

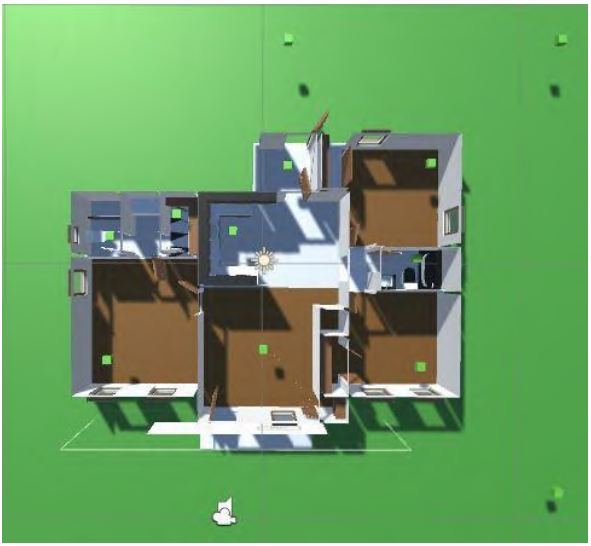


Gambar 4.7 Generate Collider Model 3D

Secara *default*, semua objek yang ada pada model 3D akan diberikan *mesh collider*. Setelah berhasil *generate collider*, model 3D diterapkan pada *scene*. Tampilan model 3D pada *scene* secara keseluruhan dapat dilihat pada Gambar 4.8 dan Gambar 4.9.



Gambar 4.8 Tampilan Model 3D Rumah Tampak Samping Depan



Gambar 4.9 Tampilan Model 3D Rumah Tampak Atas dengan Atap Terbuka

4.4. Implementasi Proses

Sub bab ini akan menjelaskan tentang implementasi dari rancangan proses yang dijelaskan pada Bab III. Penjelasan mengenai implementasi proses ini dibagi berdasarkan komponen-

komponen aplikasi. Berikut ini merupakan penjelasan dari tiap-tiap komponen.

4.4.1. Proses Melihat Menu

Proses ini merupakan proses saat menampilkan menu utama yang terdiri dari 3 tampilan yaitu menu awal, tata cara permainan, dan petunjuk peta. Implementasi proses melihat menu utama memanfaatkan fungsi "UI Canvas" dan "Button OnClick" pada Unity. Langkah-langkah implementasi proses melihat menu utama adalah pertama, buat sebuah *canvas* beserta konten dari *canvas*. Kedua, buat sebuah *EventSystem* baru untuk mendeteksi input dari pengguna. Implementasi menu utama secara keseluruhan dapat dilihat pada potongan Kode Sumber 4.1.

```
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;
using System.Collections;

public class GameController : MonoBehaviour {

    GameObject HUDPanel;
    GameObject UIMenu;
    GameObject Tutorial1;
    GameObject Tutorial2;
    GameObject UIWaiting;
    GameObject FPS;
    GameObject ExitMenu;
    SetTimer setTimer;
    Text textExitMenu;

    // Use this for initialization
    void Start () {
        HUDPanel = GameObject.Find ("HUD");
        UIMenu = GameObject.Find ("UIStartMenu");
        Tutorial1 = GameObject.Find ("UITutorial1");
        Tutorial2 = GameObject.Find ("UITutorial2");
        UIWaiting = GameObject.Find ("UIWaiting");
        FPS = GameObject.Find ("FPSController");
        ExitMenu = GameObject.Find ("UIExitMenu");
        setTimer = HUDPanel.GetComponentInChildren<SetTimer> (
);
        textExitMenu = GameObject.Find ("UIExitMenuText").
```

```
        GetComponent<Text> ();
        HUDPanel.SetActive (false);
        Tutorial1.SetActive (false);
        Tutorial2.SetActive (false);
        UIWaiting.SetActive (false);
        ExitMenu.SetActive (false);
        FPS.GetComponent <CharacterController> ().enabled = fa
lse;
        Debug.Log (HUDPanel);
    }

    // Update is called once per frame
    void Update () {

    }

    public void StartMenu() {
        print ("Game is running");
        UIMenu.SetActive (false);
        Tutorial1.SetActive (true);
    }

    public void TutorialNext1() {
        Tutorial1.SetActive (false);
        Tutorial2.SetActive (true);
    }

    public void TutorialNext2() {
        Tutorial2.SetActive (false);
        UIWaiting.SetActive (true);
        StartCoroutine ("waitingTime");
    }


    IEnumerator waitingTime ()
    {
        yield return new WaitForSeconds (5f);
        UIWaiting.GetComponentInChildren<Text> ().text =
        "Game will begin in...";
        yield return new WaitForSeconds (2f);
        UIWaiting.GetComponentInChildren<Text> ().text =
        "3...";
        yield return new WaitForSeconds (1f);
        UIWaiting.GetComponentInChildren<Text> ().text =
        "2...";
        yield return new WaitForSeconds (1f);
        UIWaiting.GetComponentInChildren<Text> ().text =
```

```
        "1...";
        yield return new WaitForSeconds (1f);
        UIWaiting.GetComponentInChildren<Text> ().text =
        "GO!";
        yield return new WaitForSeconds (1f);
        UIWaiting.SetActive (false);
        FPS.GetComponent <CharacterController> ().enabled =
        true;
        HUDPanel.SetActive (true);
    }

    public void exitGame(){
        Application.Quit ();
    }

    public void resetGame(){
        SceneManager.LoadScene (SceneManager.GetActiveScene ()
.
name);
    }

    public void GameOver()
    {
        FPS.GetComponent<CharacterController> ().enabled = fal
se;
        float minuteTime = setTimer.minute;
        float secondTime = setTimer.second;
        float fractionTime = setTimer.fraction;
        textExitMenu.text = "Your time is " +
        minuteTime.ToString("00") + " : " +
        secondTime.ToString("00") + " : " +
        fractionTime.ToString("0");
        HUDPanel.SetActive (false);
        ExitMenu.SetActive (true);
    }

}
```

Kode Sumber 4.1 Menu Utama

4.4.2. Proses Menggerakkan Karakter dengan Leap Motion

Proses ini merupakan proses menggerakkan karakter dengan Leap Motion ketika pengguna sedang menjalankan skenario. Pengguna menunjukkan tangan sesuai dengan gestur yang

ditentukan. Terdapat 5 gestur yang dapat dimanfaatkan pengguna untuk menggerakkan karakter. Gestur tangan tersebut diolah melalui data tangan yang diperoleh dari SDK Leap Motion. Gestur tangan maju dan mundur memanfaatkan sumbu z dari vektor normal telapak tangan. Gestur tangan kiri dan kanan memanfaatkan sumbu x dari vektor arah tangan. Gestur tangan berhenti memanfaatkan fungsi *PinchStrength* yang mendeteksi kekuatan tangan ketika menggengam. Implementasi dari proses ini dengan menggunakan fungsi *LateUpdate* sehingga setiap *frame* dari data tangan diolah secara *real time*. Potongan kode untuk menggerakkan karakter dengan Leap Motion dijelaskan pada Kode Sumber 4.2.

```
using UnityEngine;
using System.Collections;
using Leap;
using trinus;

public class PlayerMoveLeap : MonoBehaviour {

    Transform cameraTransform;

    public HandController handController = null;
    public float speed = 6f;

    HandModel[] hands;
    Controller controller = new Controller ();

    Vector3 movement;
    Rigidbody playerRigidbody;

    TrinusProcessor trinusProcessor;
    GameController gameController;

    // Use this for initialization
    void Start () {
        playerRigidbody = GetComponent <Rigidbody> ();
        trinusProcessor = GameObject.Find ("TrinusManager")
.GetComponent<TrinusProcessor> ();
        gameController = GameObject.Find ("GameController")
.GetComponent<GameController> ();
    }

    // Update is called once per frame
```

```
    void LateUpdate () {
        if (gameController.isStart) {
            hands = handController.GetAllPhysicsHands ();
            if (isHandEnable ()) {
                if (!isFist ()) {
                    float palmNormalZ = hands [0].GetPalmNo
rmal ().z;
                    float palmDirectionX = hands [0].GetPal
mDirection ().x;
                    Debug.Log ("NormalZ : " + palmNormalZ.T
oString () + " DirX : " + palmDirectionX.ToString ());
                    Move (palmDirectionX, palmNormalZ);
                } else
                    Debug.Log ("Player Stop");
            } else
                Debug.Log ("No Hands Detected");
        }
    }

    void Move (float h, float v)
    {
        // Set the movement vector based on the axis input.
        if (cameraTransform == null && trinusProcessor.getM
ainCamera () != null) {
            cameraTransform = trinusProcessor.getMainCamera
 ().transform;
        }
        if (cameraTransform != null)
            movement = GetVectorRelativeToObject (new Vecto
r3 (h, 0f, v), cameraTransform);

        // Normalise the movement vector and make it propor
tional to the speed per second.
        movement = movement.normalized * speed * Time.delta
Time;

        // Move the player to it's current position plus th
e movement.
        playerRigidbody.MovePosition (transform.position +
movement);
    }

    public static Vector3 GetVectorRelativeToObject(Vector3
 inputVector, Transform camera)
    {
        Vector3 objectRelativeVector = Vector3.zero;
        if (inputVector != Vector3.zero)
        {
            Vector3 forward = camera.TransformDirection(Vec
```

```
tor3.forward);
            forward.y = 0f;
            forward.Normalize();
            Vector3 right = new Vector3(forward.z, 0.0f, -
forward.x);

            Vector3 relativeRight = inputVector.x * right;
            Vector3 relativeForward = inputVector.z * forwa
rd;

            objectRelativeVector = relativeRight + relative
Forward;

            if (objectRelativeVector.magnitude > 1f) object
RelativeVector.Normalize();
        }
        return objectRelativeVector;
    }

    bool isHandEnable () {
        if (hands.Length > 0) {
            return true;
        } else
            return false;
    }

    bool isFist() {
        Frame frame = controller.Frame ();
        HandList hands = frame.Hands;
        Hand firstHand = hands [0];

        float pinch = firstHand.PinchStrength;

        if (pinch >= 0.5) {
            return true;
        } else
            return false;
    }
}
```

Kode Sumber 4.2 Menggerakkan Karakter dalam Leap Motion

4.4.3. Proses Melihat *Checkpoint*

Proses ini merupakan proses yang berjalan ketika pengguna mulai menjalankan skenario. *Checkpoint* dibuat dari objek "Cube" yang didalamnya terdapat *Box Collider* sehingga dapat

menjalankan aksi ketika diberi *trigger*. Pada karakter yang digerakkan oleh pengguna, terdapat fungsi *PlayerCheck* yang menginisialisasi *checkpoint* pada skenario. Potongan kode dari fungsi *PlayerCheck* dijelaskan pada Kode Sumber 4.3.

```
using UnityEngine;
using System.Collections;

public class PlayerCheck : MonoBehaviour {

    public Transform[] checkPointArray;
    public static Transform[] checkpointA;
    public static int currentCheckpoint = 0;
    public Vector3 startPos;
    public GameObject HUDTextCheckpointChange;
    public static int minute;
    public static int second;
    public static int fraction;
    private GameController gameController;

    // Use this for initialization
    void Start () {
        GameObject gameControllerObject = GameObject
.Find ("GameController");
        gameController = gameControllerObject.GetCom
ponent<GameController> ();
        startPos = transform.position;
        currentCheckpoint = 0;
        HUDTextCheckpointChange = GameObject.Find ("
HUDTextCheckpoint");

    }

    // Update is called once per frame
    void Update () {
        if (currentCheckpoint < checkPointArray.Leng
th) {
            checkpointA = checkPointArray;
            HUDTextCheckpointChange.GetComponent<Tex
tMesh> ().text = "Checkpoint " + currentCheckpoint.T
oString () + " / 12";
        } else
            return;
```

```
    }

    void OnTriggerEnter(Collider other)
    {
        if (other.tag == "Finish") {
            gameController.GameOver ();
        }
    }
}
```

Kode Sumber 4.3 Fungsi *PlayerCheck*

Dalam objek "Cube" terdapat fungsi *CheckPoint* yang berguna untuk menentukan perilaku objek layaknya sebuah *checkpoint*. Fungsi *CheckPoint* memanfaatkan *OnTriggerEnter* untuk menjalankan suatu tugas ketika suatu objek telah bersinggungan dengan objek lain. Potongan kode dari fungsi *CheckPoint* dapat dilihat pada Kode Sumber 4.4.

```
using UnityEngine;
using System.Collections;

public class CheckPoint : MonoBehaviour {


    // Use this for initialization
    void Start () {

    }

    void Update ()
    {
        RotateBox ();

    }

    void RotateBox ()
    {
        this.transform.Rotate(Vector3.up * Time.delt
aTime * 50);
    }
```

```
    void OnTriggerEnter (Collider col)
    {
        if (!col.CompareTag ("Player"))
            return;

        if (transform == PlayerCheck.checkpointA [Pl
ayerCheck.currentCheckpoint].transform)
        {
            if (PlayerCheck.currentCheckpoint + 1 <
PlayerCheck.checkpointA.Length) {
                int newCheckpoint = PlayerCheck.curr
entCheckpoint + 1;
                Debug.Log ("Checkpoint " + newCheckp
oint + " reached");
                PlayerCheck.currentCheckpoint++;
                Destroy (this.gameObject);
            }
            else
            {
                Destroy (this.gameObject);
                Debug.Log ("Finish");
                GameObject.Find ("HUDTextTime").GetC
omponent<SetTimer> ().count = true;
            }
        }
    }


}
```

Kode Sumber 4.4 Fungsi CheckPoint

4.4.4. Proses Mencatat Waktu Tempuh

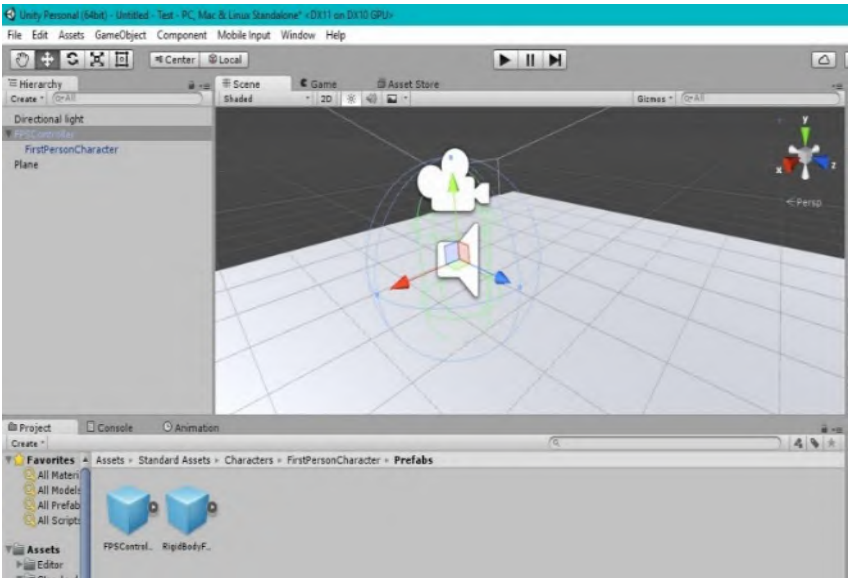
Proses ini merupakan proses yang berjalan bersamaan dengan proses melihat *checkpoint*. Proses ini mencatat waktu yang ditempuh oleh pengguna ketika mulai menjalankan skenario hingga pengguna selesai menempuh semua *checkpoint*. Waktu yang berjalan ditampilkan pada *first person view*. Potongan kode untuk mencatat waktu tempuh dapat dijelaskan pada Kode Sumber 4.5.

```
using UnityEngine;
using System.Collections;

public class SetTimer : MonoBehaviour {

    public bool count;
    public int minute;
    public int second;
    public int fraction;
    float timecount;
    float starttime;
    // Use this for initialization
    void Start () {
        //starttime = Time.time;
    }

    // Update is called once per frame
    void Update () {
        if (!count) {
            StartTime ();
        } else
            return;

    }

    void StartTime() {
        timecount += Time.deltaTime;
        minute = Mathf.FloorToInt(timecount / 60f);
        second = Mathf.FloorToInt(timecount % 60f);
        fraction = Mathf.FloorToInt((timecount * 10)
 % 10);
        this.GetComponent<TextMesh> ().text = "Time
: " + minute.ToString("00") + " : " + second.ToStrin
g("00") + " : " + fraction.ToString("0") ;
    }
}
```

Kode Sumber 4.5 Fungsi Mencatat Waktu Tempuh

4.5. Implementasi Pergerakan Karakter dengan *Gamepad*

Unity telah menyediakan *standard assets* untuk menggerakkan karakter dengan menggunakan *gamepad*. Untuk memanfaatkan *assets* tersebut, maka pilih menu *assets -> import package -> Characters* pada Unity. Unity akan mengimpor isi dari *standard assets* ke *project* yang sedang dibuat. Setelah itu, pilih *prefabs FPSController* dan tarik ke dalam *scene* untuk menampilkan karakter. Secara *default*, karakter dapat digerakkan dengan *gamepad*. Hasil proses tersebut dapat dilihat pada Gambar 4.10.

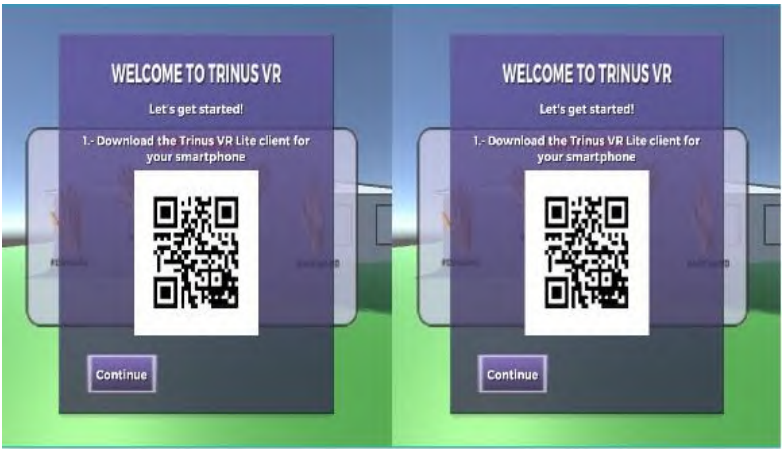


Gambar 4.10 Implementasi Standard Assets pada Unity

4.6. Implementasi Integrasi Trinus VR pada Unity

Untuk menghubungkan perangkat komputer dengan *smartphone*, diperlukan SDK Trinus VR yang diimplementasikan pada Unity. Penggunaan Trinus VR digunakan pada interaksi dengan menggunakan Leap Motion. Terdapat beberapa langkah untuk mengintegrasikan SDK Trinus VR pada Unity. Pertama,

unduh SDK Trinus VR melalui *Assets Store* Unity. Kedua, impor SDK Trinus VR ke dalam *project* yang dibuat. Terdapat beberapa *file* yang diperoleh setelah mengimpor SDK Trinus VR. Ketiga, pilih *prefab TrinusCamera, TrinusManager, TrinusUI,* dan *DefaultEventSystem* kemudian bawa *prefab* tersebut ke dalam *scene*. Tambahkan pula *SampleTrinusOnlyUIManager* sebagai *default* dari tampilan antarmuka dari Trinus VR pada aplikasi. Pasangkan *TrinusCamera* pada sebuah *capsule* supaya karakter memiliki sebuah *collider* sehingga karakter dapat mendeteksi adanya benda lain yang memiliki *collider*. Tampilan antar muka Trinus VR pada aplikasi dapat dilihat pada Gambar 4.11.

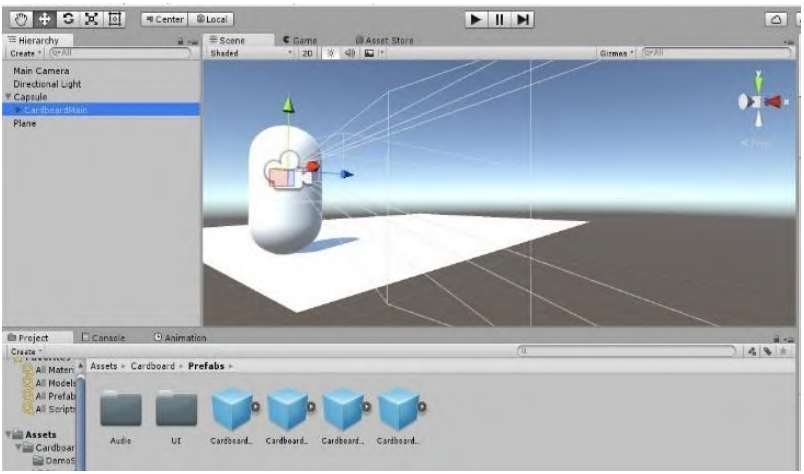


Gambar 4.11 Tampilan Antarmuka Trinus VR

4.7. Implementasi Integrasi Google Cardboard pada Unity

Google Cardboard membutuhkan konfigurasi tampilan antarmuka khusus agar dapat mengikuti lengkungan dari lensa. Integrasi tersebut dilakukan dengan menambahkan SDK Google Cardboard pada Unity. SDK Google Cardboard digunakan untuk

interaksi dengan *gamepad*. SDK Google Cardboard dapat diunduh melalui situs https://developers.google.com/vr/unity/. Melalui situs tersebut, didapat sebuah *package* yang dapat diimpor kedalam Unity. Setelah proses impor selesai, maka akan muncul beberapa folder baru pada *Assets*. Untuk menerapkan kamera Google Cardboard sebagai kamera utama dalam *scene* di Unity, maka digunakan *CardboardMain* yang ada pada folder *prefab*. *CardboardMain* ditaruh pada karakter yang akan digerakkan untuk menciptakan tampilan *first person* yang *immersive*. Hasil proses tersebut dapat dilihat pada Gambar 4.12.

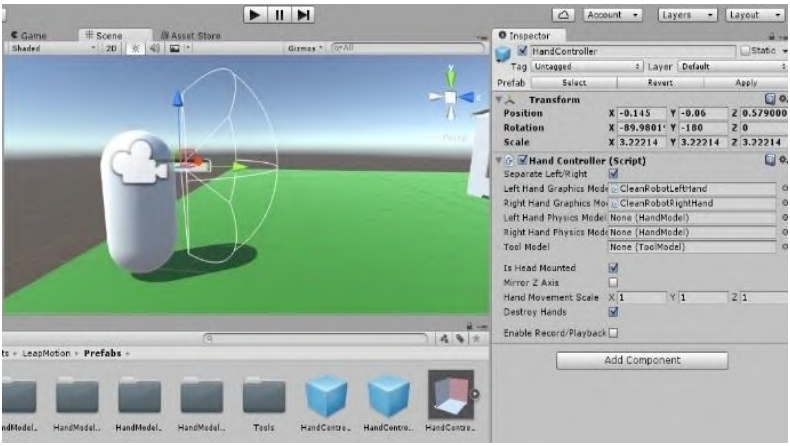


Gambar 4.12 Implementasi Google Cardboard SDK pada Unity

4.8. Implementasi Integrasi Leap Motion pada Unity

Integrasi Leap Motion dengan Unity memerlukan suatu penghubung yaitu SDK Leap Motion. SDK Leap Motion dapat diunduh melalui situs resmi Leap Motion yaitu https://developer.leapmotion.com/. *Package* yang berhasil diunduh dari situs tersebut kemudian diimpor kedalam Unity. Setelah proses impor berhasil, maka akan muncul empat buah folder baru pada bagian *Assets* di Unity.

Untuk implementasi sensor area Leap Motion, maka digunakan *HandController* yang ada di dalam *prefab* Leap Motion. *HandController* dimasukan ke dalam *scene* Unity. Posisi *HandController* diputar 90 derajat pada sumbu x dan 270 derajat pada sumbu y sehingga posisi alat mudah dijangkau oleh tangan ketika menempel pada Google Cardboard. Hasil proses tersebut dapat dilihat pada Gambar 4.13.



Gambar 4.13 Implementasi HandController Leap Motion pada Unity

[Halaman ini sengaja dikosongkan]

BAB V
PENGUJIAN DAN EVALUASI

Bab ini membahas pengujian dan evaluasi pada aplikasi yang dikembangkan. Pengujian yang dilakukan adalah pengujian interaksi dan kuesioner pasca pengujian. Metode pengujian telah dijelaskan pada Metodologi Pengujian dan Evaluasi pada Bab I. Hasil evaluasi menjabarkan tentang rangkuman hasil pengujian pada bagian akhir bab ini.

5.1. Lingkungan Pengujian

Lingkungan pengujian sistem pada pengerjaan Tugas Akhir ini dilakukan pada lingkungan dan alat kakas seperti yang tertera pada Tabel 5.1.

Tabel 5.1 Lingkungan Pengujian Sistem

Perangkat Keras	- Laptop Asus A43SD
	  o Prosesor Intel(R) Core(TM) i5-2450M CPU @ 3.30GHz
	  o RAM 8 GB
	- Google Cardboard
	- Leap motion Controller
	- *Gamepad*
	- *Smartphone* Sony Xperia Z2
	  o Prosesor Qualcomm Snapdragon 801@2.3 GHz Quad-core
	  o RAM 3 GB

Perangkat Lunak	- Laptop Asus A43SD
	  o Sistem Operasi Microsoft Windows 10 Pro 64-bit
	- *Smartphone* Sony Xperia Z2
	  o *Sistem Operasi Android*

5.2. Pengujian Fungsionalitas

Pengujian fungsionalitas dilakukan dengan menyiapkan beberapa skenario pengujian sebagai tolok ukur keberhasilan pengujian dan mengacu pada kasus penggunaan yang sudah dideskripsikan pada Bab 3.2.2. Metode pengujian yang digunakan pada pengujian fungsionalitas mengacu pada *whitebox testing* dan *blackbox testing*.

5.2.1. Pengujian Menggerakkan Karakter dengan Leap Motion

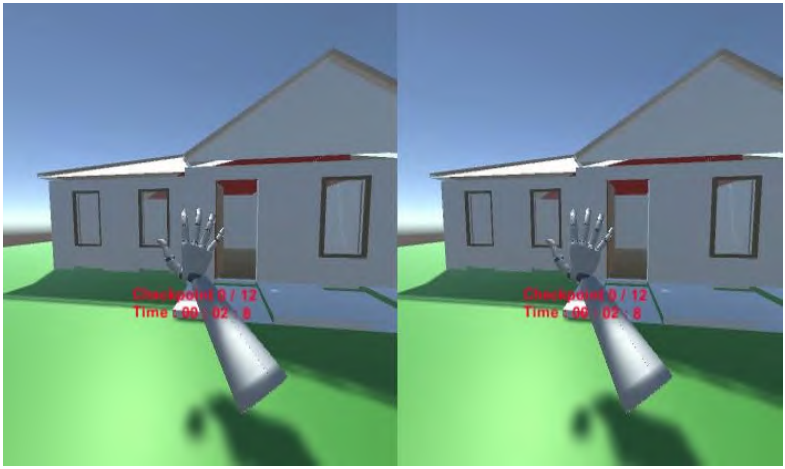
Pengujian menggerakkan karakter dengan Leap Motion merupakan pengujian terhadap aplikasi untuk mengetahui ketepatan gerakan karakter sesuai dengan gestur tangan yang dilakukan oleh pengguna melalui Leap Motion. Terdapat 6 skenario percobaan yang diujikan. Tiap skenario diujikan sebanyak 10 kali percobaan selama 3 detik. Hasil dari pengujian menggerakkan karakter dengan Leap Motion dapat dilihat pada Tabel 5.2.

Tabel 5.2 Pengujian menggerakkan karakter dengan Leap Motion

ID	SP-UC-001
Referensi Kasus Penggunaan	UC-001
Nama	Menggerakkan Karakter dengan Leap Motion
Tujuan	Mengecek apakah aplikasi dapat menggerakkan karakter dalam lingkungan virtual sesuai dengan gestur tangan yang telah ditentukan serta sesuai dengan arah pandangan kamera dengan Google Cardboard
Kondisi Awal	Pengguna sudah selesai membuka menu utama

ID	SP-UC-001
Skenario 1	**Pengguna diminta untuk menghadapkan tangan ke depan**
Keluaran yang Diharapkan	Aplikasi dapat menggerakkan karakter ke depan
Hasil Pengujian	9 dari 10 percobaan berhasil
Skenario 2	**Pengguna diminta untuk menghadapkan tangan ke belakang**
Keluaran yang Diharapkan	Aplikasi dapat menggerakkan karakter ke belakang
Hasil Pengujian	10 dari 10 percobaan berhasil
Skenario 3	**Pengguna diminta untuk menghadapkan tangan ke depan dan condongkan ke kiri**
Keluaran yang Diharapkan	Aplikasi dapat menggerakkan karakter ke kiri
Hasil Pengujian	8 dari 10 percobaan berhasil
Skenario 4	**Pengguna diminta untuk menghadapkan tangan ke depan dan condongkan ke kanan**
Keluaran yang Diharapkan	Aplikasi dapat menggerakkan karakter ke kanan
Hasil Pengujian	9 dari 10 percobaan berhasil
Skenario 5	**Pengguna diminta untuk mengubah arah padangan kamera Google Cardboard berbeda dari keadaan awal kemudian mengarahkan tangan ke depan**
Keluaran yang Diharapkan	Aplikasi dapat menggerakkan karakter ke depan sesuai dengan arah pandangan kamera Google Cardboard
Hasil Pengujian	9 dari 10 percobaan berhasil

ID	SP-UC-001
Skenario 6	**Pengguna diminta untuk menghadapkan tangan ke depan kemudian mengepalkan tangan**
Keluaran yang Diharapkan	Aplikasi dapat memberhentikan karakter setelah menggerakkan karakter
Hasil Pengujian	9 dari 10 percobaan berhasil



Gambar 5.1 Tampilan pengguna saat pengujian menggerakkan karakter dengan Leap Motion

Terdapat beberapa kendala selama pengujian berlangsung. Pertama, Leap Motion kadang tidak dapat mendeteksi tangan dengan tepat ketika tempat di sekitar Leap Motion terdapat banyak gangguan. Kedua, ketika karakter bergerak ke kiri dan kanan, karakter begerak secara diagonal. Hal ini mengakibatkan karakter tidak bergerak ke kiri atau kanan.

5.2.2. Pengujian Menggerakkan Karakter dengan *Gamepad*

Pengujian menggerakkan karakter dengan *gamepad* merupakan pengujian terhadap aplikasi untuk mengetahui ketepatan gerakan karakter sesuai dengan masukan dari *gamepad*. Terdapat 5 skenario yang diujikan. Hasil dari pengujian menggerakkan karakter dengan *gamepad* dapat dilihat pada Tabel 5.3.

Tabel 5.3 Pengujian menggerakkan karakter dengan *gamepad*

ID	SP-UC-002
Referensi Kasus Penggunaan	UC-002
Nama	Menggerakkan Karakter dengan *gamepad*
Tujuan	Mengecek apakah aplikasi dapat menggerakkan karakter dalam lingkungan virtual sesuai dengan masukan dari *gamepad* serta sesuai dengan arah pandangan kamera dengan Google Cardboard
Kondisi Awal	Pengguna sudah selesai membuka menu utama
Skenario 1	**Pengguna diminta untuk menggerakkan *analog gamepad* ke atas**
Keluaran yang Diharapkan	Aplikasi dapat menggerakkan karakter ke depan
Hasil Pengujian	Berhasil
Skenario 2	**Pengguna diminta untuk menggerakkan *analog gamepad* ke bawah**
Keluaran yang Diharapkan	Aplikasi dapat menggerakkan karakter ke belakang
Hasil Pengujian	Berhasil
Skenario 3	**Pengguna diminta untuk menggerakkan *analog gamepad* ke kiri**

ID	SP-UC-002
Keluaran yang Diharapkan	Aplikasi dapat menggerakkan karakter ke kiri
Hasil Pengujian	Berhasil
Skenario 4	**Pengguna diminta untuk menggerakkan *analog gamepad* ke kanan**
Keluaran yang Diharapkan	Aplikasi dapat menggerakkan karakter ke kanan
Hasil Pengujian	Berhasil
Skenario 5	**Pengguna diminta untuk mengubah arah padangan kamera Google Cardboard berbeda dari keadaan awal kemudian mengarahkan tangan ke depan**
Keluaran yang Diharapkan	Aplikasi dapat menggerakkan karakter ke depan sesuai dengan arah pandangan kamera Google Cardboard
Hasil Pengujian	Berhasil

5.2.3. Pengujian Menampilkan *Checkpoint*

Pengujian menampilkan *checkpoint* merupakan pengujian terhadap aplikasi untuk memastikan *checkpoint* yang harus dilewati pengguna telah sesuai dengan kebutuhan. Terdapat 2 skenario yang diujikan. Hasil dari pengujian menampilkan *checkpoint* dapat dilihat pada Tabel 5.4.

Tabel 5.4 Hasil pengujian menampilkan *checkpoint*

ID	SP-UC-003
Referensi Kasus Penggunaan	UC-003
Nama	Menampilkan *checkpoint*

ID	SP-UC-003
Tujuan	Mengecek apakah aplikasi dapat menampilkan *checkpoint* dengan benar dan berjalan sesuai dengan kebutuhan
Kondisi Awal	Pengguna sudah selesai membuka menu utama
Skenario 1	**Pengguna diminta untuk melewati seluruh *checkpoint* sesuai dengan urutan**
Keluaran yang Diharapkan	Aplikasi dapat menampilkan seluruh *checkpoint*, menghapus *checkpoint* yang sudah terlewati, serta memperbarui indikator *checkpoint* pada layar utama.
Hasil Pengujian	Berhasil
Skenario 2	**Pengguna diminta untuk melewati seluruh *checkpoint* tidak sesuai dengan urutan**
Keluaran yang Diharapkan	Aplikasi dapat mengetahui bahwa *checkpoint* yang dilewati tidak sesuai dengan urutan dan tidak menghapus *checkpoint* yang dilewati
Hasil Pengujian	Berhasil

5.2.4. Pengujian Mencatat Waktu Tempuh

Pengujian mencatat waktu tempuh merupakan pengujian terhadap aplikasi untuk memastikan bahwa pencatatan waktu tempuh dapat berjalan secara tepat selama pengguna menjalankan permainan. Terdapat 2 skenario yang diujikan. Hasil dari pengujian mencatat waktu tempuh dapat dilihat pada Tabel 5.5.

Tabel 5.5 Hasil pengujian mencatat waktu tempuh

ID	SP-UC-004
Referensi Kasus Penggunaan	UC-004
Nama	Mencatat waktu tempuh
Tujuan	Mengecek apakah aplikasi dapat mencatat waktu tempuh dengan benar
Kondisi Awal	Pengguna sudah selesai membuka menu utama
Skenario 1	**Pengguna diminta untuk menjalankan permainan**
Keluaran yang Diharapkan	Aplikasi dapat menampilkan waktu pada layar utama dan waktu dimulai tepat setelah permainan berjalan
Hasil Pengujian	Berhasil
Skenario 2	**Pengguna diminta untuk melewati seluruh *checkpoint* sesuai dengan urutan**
Keluaran yang Diharapkan	Aplikasi dapat menampilkan menu akhir yang berisikan waktu tempuh pengguna
Hasil Pengujian	Berhasil

5.3. Pengujian Interaksi

Pengujian interaksi dilakukan untuk mengetahui efektivitas suatu interaksi pada lingkungan virtual. Berikut ini penjabaran skenario dan hasil uji coba yang dilakukan terhadap perangkat lunak yang dibangun.

5.3.1. Skenario Pengujian Interaksi

Tujuan utama pengujian interaksi ini adalah untuk mengetahui kecepatan pengguna dalam menggunakan *input*

sebagai interaksi pada lingkungan virtual. Terdapat dua jenis *input* yang akan diujicoba, yaitu Leap Motion Controller dan *gamepad*. Skenario uji coba dilakukan pada Google Cardboard. Skenario uji coba yang dilakukan yaitu pengguna diminta untuk melewati 12 *checkpoint* yang sudah ditentukan letaknya. Parameter utama dari uji coba ini adalah waktu tempuh penyelesaian skenario. Untuk mengetahui adaptasi pengguna dalam memakai *input*, pengguna menjalankan 3 kali pengujian pada setiap *input*.

5.3.2. Cara Menjalankan Aplikasi

Terdapat 2 cara dalam menjalankan aplikasi untuk melakukan pengujian pada masing-masing interaksi. Ada beberapa tahapan sebelum pengguna dapat menjalankan aplikasi.

5.3.2.1. Interaksi dengan Leap Motion

Sebelum pengguna dapat menjalankan aplikasi interaksi dengan Leap Motion, ada beberapa tahapan yang harus dilakukan. Berikut tahapan yang harus dilakukan :

1. Pasang aplikasi Leap Motion pada laptop. Aplikasi Leap Motion dapat diunduh melalui situs https://www.leapmotion.com/setup.
2. Pasang aplikasi Trinus VR pada *smartphone*. Aplikasi dapat diunduh melalui Play Store di Android.
3. Pasangkan Leap Motion pada Google Cardboard. Sambungkan Leap Motion ke laptop. Contoh pemasangan Leap Motion pada Google Cardboard dapat dilihat pada Gambar
4. Aktifkan *wifi* dari laptop dan *smartphone*, kemudian sambungkan ke jaringan yang sama.
5. Jalankan aplikasi pengujian pada laptop. Ikuti langkah-langkah yang diberikan.
6. Jalankan aplikasi Trinus VR pada *smartphone* dan tekan tombol “start” untuk menyambungkan perangkat.
7. Letakkan *smartphone* dalam Google Cardboard

5.3.2.2. Interaksi dengan *Gamepad*

Sebelum pengguna dapat menjalankan aplikasi interaksi dengan *gamepad*, ada beberapa tahapan yang harus dilakukan. Berikut tahapan yang harus dilakukan :

1. Pasang aplikasi pengujian pada *smartphone*
2. Sambungkan *Bluetooth gamepad* pada *smartphone*
3. Jalankan aplikasi pengujian
4. Letakkan *smartphone* dalam Google Cardboard

5.3.3. Peserta Pengujian Interaksi

Seluruh pengujian yang dilakukan melibatkan 6 mahasiswa Teknik Informatika ITS yang terdiri dari 3 laki-laki dan 3 perempuan. Sebelum penguji menjalankan pengujian, penguji diminta untuk mengisi kuesioner singkat yang berhubungan dengan pengujian. Daftar nama penguji yang terlibat serta hasil kuesioner pra-pengujian dapat dilihat pada Tabel 5.6.

Tabel 5.6 Daftar Nama Penguji

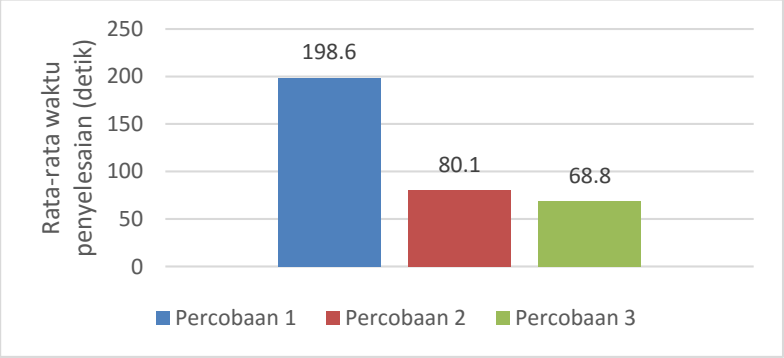
No.	Nama Penguji	Pengalaman Realitas Virtual	Pengalaman *gamepad*
1	Aditya Putra F.	✓	✓
2	Ignatius Abraham S.	✓	✓
3	Putu Adhi Purwanto	✓	✓
4	Bima Nisrina M.	✓	✓
5	Maulina Nur Istiqomah		✓
6	Diagnosa F.	✓	

5.3.4. Hasil Pengujian Interaksi

Sub bab ini menjelaskan hasil yang didapatkan setelah pengujian yang dilakukan pada pengguna. Penjelasan terdiri dari penjabaran waktu yang ditempuh oleh pengguna ketika menjalankan skenario yang dilakukan.

5.3.4.1. Pengujian Interaksi dengan *Gamepad*

Uji coba interaksi dengan *gamepad* ini bertujuan memperoleh waktu tempuh pengguna untuk menyelesaikan skenario yang diberikan ketika pengguna menggunakan *gamepad.* Hasil uji coba interaksi dengan *gamepad* dapat dilihat secara rinci pada Gambar 5.2.



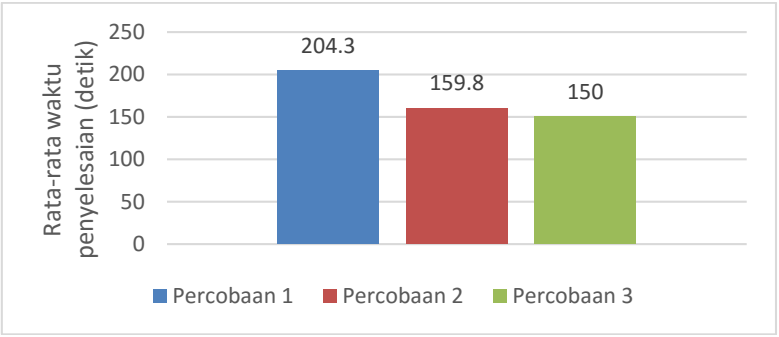
Gambar 5.2 Grafik Rekap Waktu Penyelesaian Uji Coba Interaksi dengan Gamepad

Hasil pencatatan waktu penyelesaian uji coba interaksi dengan *gamepad* dipengaruhi oleh beberapa kendala selama pengujian. Pertama, penguji masih beradaptasi dengan kontrol pada *gamepad* ketika menggunakan realitas virtual. Hal ini disebabkan karena kebiasaan pengguna ketika menggunakan *gamepad* di luar realitas virtual berbeda ketika di dalam realitas virtual. Kedua, beberapa penguji merasa pusing ketika memakai Google Cardboard. Hal ini disebabkan karena sewaktu

menggerakkan karakter, penguji harus memutar arah Google Cardboard sehingga banyak gerakan yang harus dilakukan.

5.3.4.2. Pengujian Interaksi dengan Leap Motion

Uji coba interaksi dengan Leap Motion ini bertujuan memperoleh waktu tempuh pengguna untuk menyelesaikan skenario yang diberikan ketika pengguna menggunakan Leap Motion. Hasil uji coba interaksi dengan Leap Motion dapat dilihat secara rinci pada Gambar 5.3.



Gambar 5.3 Grafik Rekap Waktu Penyelesaian Uji Coba Interaksi dengan Leap Motion

Hasil pencatatan waktu penyelesaian uji coba interaksi dengan Leap Motion dipengaruhi oleh beberapa kendala selama pengujian. Pertama, penguji masih belum familiar dengan interaksi menggunakan Leap Motion. Kedua, aplikasi berjalan kurang lancar selama pengujian. Hal itu disebabkan oleh jaringan yang dipakai serta spesifikasi dari perangkat keras yang dipakai. Ketiga, sering kali Leap Motion salah mendeteksi tangan. Selama pengujian, penguji mengalami kesulitan karena tiba-tiba muncul tangan diluar jangkauan. Hal ini disebabkan oleh kurang sterilnya lingkungan pengujian dari benda-benda yang terdeteksi salah sebagai tangan oleh Leap Motion.

5.4. Kuesioner Interaksi

Pada subbab ini dijelaskan hasil kuesioner interaksi yang diisi oleh pengguna setelah melakukan skenario uji coba interaksi. Terdapat 2 jenis kuesioner yang dilakukan yaitu kuesioner pengujian interaksi serta kritik dan saran.

Pada kuesioner pengujian itneraksi, pengguna melakukan penilaian untuk setiap interaksi yang telah diujikan pada pengujian interaksi. Setiap poin penilaian memiliki skala nilai 1 sampai dengan 6 dengan keterangan 1 adalah sangat tidak setuju, 2 adalah tidak setuju, 3 adalah kurang setuju, 4 adalah cukup setuju, 5 adalah setuju, dan 6 adalah sangat setuju.

5.4.1. Kuesioner Interaksi *Gamepad*

Setelah kuesioner dilakukan, keseluruhan nilai modus dari kuesioner interaksi dengan *gamepad* dapat dilihat pada 5.7.

Tabel 5.7 Hasil Kuesioner Interaksi dengan Gamepad

No.	Parameter	Nilai Modus
1	[Akurasi] Saya merasa interaksi yang digunakan membantu mencapai target	5
2	[*Ease of Learning*] Saya merasa interaksi yang digunakan mudah dipelajari	5
3	[Kehadiran] Saya merasa berada di lokasi sesungguhnya ketika menggunakan interaksi ini	5
4	[*Ease of Use*] Saya merasa tidak kesulitan dalam menggunakan interaksi ini	5
5	[Kenyamanan] Saya merasa nyaman ketika menggunakan interaksi ini	3

Secara keseluruhan penguji setuju penggunaan interaksi dengan *gamepad* membantu penguji mencapai target dan mudah untuk dipelajari serta dipakai. Selain itu, penguji merasa berada di lingkungan virtual ketika menggunakan interaksi *gamepad*.

Namun, penguji masih merasa kurang nyaman ketika menggunakan *gamepad*.

5.4.2. Kuesioner Interaksi Leap Motion

Setelah kuesioner dilakukan, keseluruhan nilai modus dari kuesioner interaksi dengan *gamepad* dapat dilihat pada Tabel 5.8.

Tabel 5.8 Hasil Kuesioner Interaksi dengan Leap Motion

No.	Parameter	Nilai Modus
1	[Akurasi] Saya merasa interaksi yang digunakan membantu mencapai target	4
2	[*Ease of Learning*] Saya merasa interaksi yang digunakan mudah dipelajari	5
3	[Kehadiran] Saya merasa berada di lokasi sesungguhnya ketika menggunakan interaksi ini	4
4	[*Ease of Use*] Saya merasa tidak kesulitan dalam menggunakan interaksi ini	4
5	[Kenyamanan] Saya merasa nyaman ketika menggunakan interaksi ini	3

Penguji merasa interaksi dengan Leap Motion mudah untuk dipelajari. Interaksi dengan Leap Motion dirasa cukup membantu mencapai target, dan relatif mudah untuk digunakan. Penguji juga merasa berada di lingkungan virtual. Namun, penguji masih merasa kurang nyaman ketika menggunakan Leap Motion.

5.4.3. Kritik dan Saran

Setelah mengisi kuesioner pengujian, penguji diminta untuk memberikan saran dan kritiknya selama pengujian. Saran dan kritik berguna sebagai evaluasi yang diberikan oleh penguji untuk pengembangan selanjutnya. Detil saran dan kritik yang diberikan oleh penguji dapat dilihat pada Tabel 5.9.

Tabel 5.9 Kritik dan saran dari penguji

Nama Penguji	Saran dan Kritik
Aditya Putra F.	Pengambilan *checkpoint* diberi tanda panah pada peta. Performa *hardware* Leap Motion dan Laptop ditingkatkan agar tidak *lag.*
Ignatius Abraham S.	Untuk tempat pengujian adalah tempat yang bebas gangguan.
Putu Adhi Purwanto	Kadang tembus tembok/atap, pergerakan menggunakan Leap Motion kurang pelan.
Bima Nisrina M.	Gestur belok dan mundurnya kurang mudah digunakan
Maulina Nur Istiqomah	Temboknya bisa tembus-tembus (di kamar mandi terutama), pusing pas dipake
Diagnosa Fenomena	Temboknya beberapa masih tembus. Untuk *control* Leap Motion maju dan mundurnya terlalu cepat.

5.5. Evaluasi Pengujian

Berdasarkan hasil pengujian fungsionalitas, pengujian interaksi dan kuesioner interaksi yang telah dijelaskan pada subbab sebelumnya, maka didapatkan evaluasi sebagai berikut.

5.5.1. Evaluasi Pengujian Fungsionalitas

Evaluasi pengujian fungsionalitas dilakukan dengan menampilkan data rekapitulasi perangkat lunak yang telah dipaparkan pada subbab 5.2. Dalam hal ini, rekapitulasi disusun dalam bentuk tabel yang dapat dilihat pada Tabel 5.10. Dari data yang terdapat pada tabel tersebut, diketahui bahwa aplikasi yang dibuat telah memenuhi kasus penggunaan yang telah ditentukan.

Tabel 5.10 Rekapitulasi Hasil Pengujian Fungsionalitas

ID	Deskripsi		Hasil
SP-UC-001	Menggerakkan karakter dengan Leap Motion	Skenario 1	9/10 Berhasil
		Skenario 2	10/10 Berhasil
		Skenario 3	8/10 Berhasil
		Skenario 4	9/10 Berhasil
		Skenario 5	9/10 Berhasil
		Skenario 6	9/10 Berhasil
SP-UC-002	Menggerakkan karakter dengan *gamepad*	Skenario 1	Berhasil
		Skenario 2	Berhasil
		Skenario 3	Berhasil
		Skenario 4	Berhasil
		Skenario 5	Berhasil
SP-UC-001	Menampilkan *checkpoint*	Skenario 1	Berhasil
		Skenario 2	Berhasil
SP-UC-001	Mencatat waktu tempuh	Skenario 1	Berhasil
		Skenario 2	Berhasil

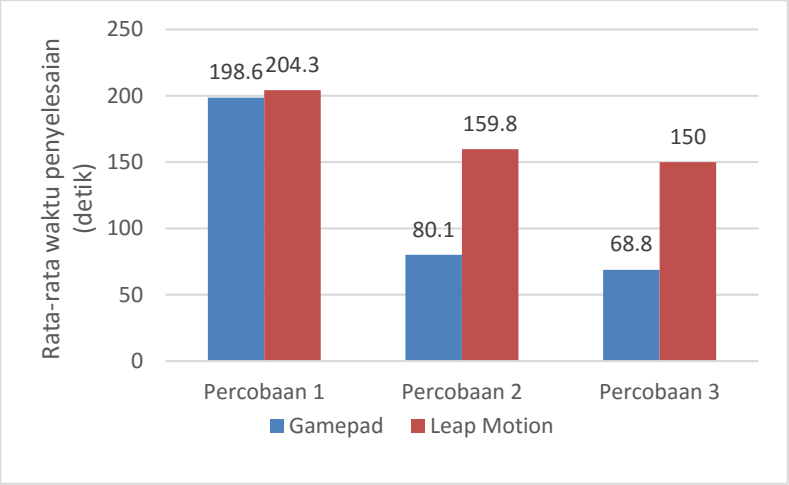
5.5.2. Evaluasi Pengujian Interaksi

Evaluasi pengujian interaksi dilakukan dengan menampilkan data rekapitulasi hasil pengujian interaksi yang sudah dipaparkan pada bab 5.3.4. Setelah skenario pengujian interaksi dijalankan pada masing-masing interaksi, hasil rekapitulasi pencatatan waktu masing-masing interaksi dapat dilihat pada Gambar 5.4.

Melalui hasil tersebut, peningkatan waktu yang diperoleh dari perobaan 1 ke percobaan 3 pengujian interaksi dengan

gamepad mencapai 65.36%, sedangkan peningkatan waktu yang diperoleh dari percobaan 1 ke percobaan 3 pengujian interaksi dengan Leap Motion mencapai 26,58%. Melalui perbandingan tersebut, dapat disimpulkan bahwa interaksi dengan *gamepad* lebih mudah diadaptasi dibandingkan interaksi dengan Leap Motion.

Salah satu faktor yang menyebabkan mudahnya adaptasi penggunaan *gamepad* pada lingkungan virtual yaitu 5 dari 6 penguji sudah pernah menggunakan *gamepad* sebelumnya. Peningkatan waktu pengguna ketika menggunakan interaksi *gamepad* sudah terlihat dari percobaan 1 ke percobaan 2.



Gambar 5.4 Perbandingan Hasil Rekap Waktu Pengujian Interaksi

5.5.3. Evaluasi Kuesioner Interaksi

Evaluasi kuesioner interaksi dilakukan dengan mengisi lembar pengujian oleh 6 penguji interaksi. Melalui kuesioner interaksi, diperoleh hasil seperti yang disimpulkan pada Tabel 5.11.

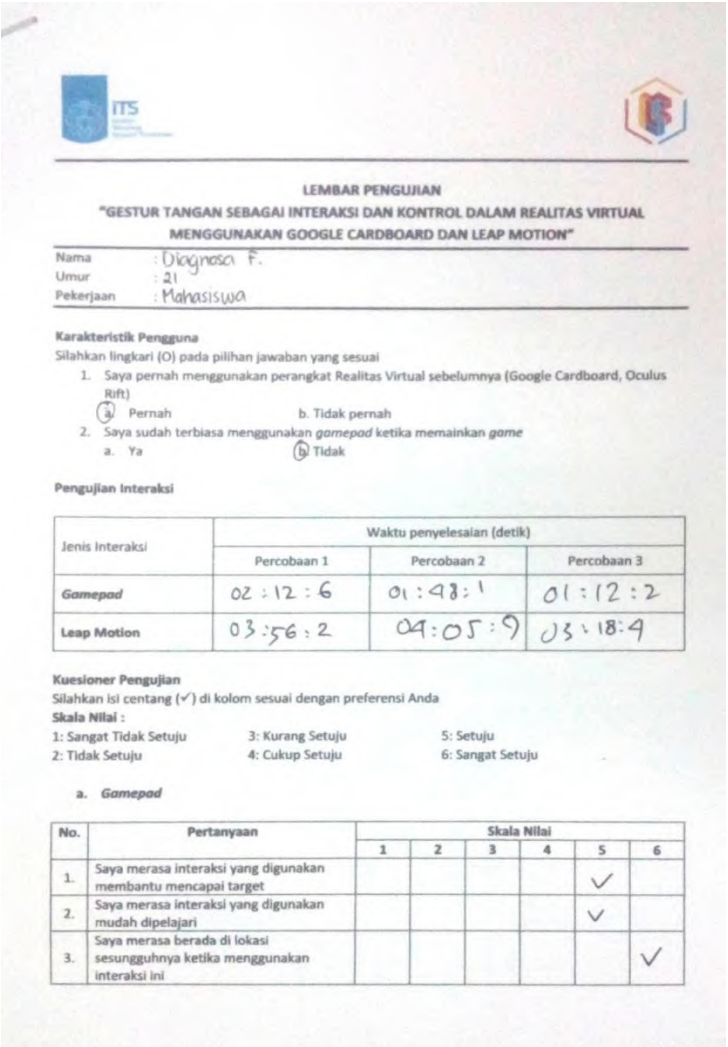
Tabel 5.11 Rekap Hasil Kuesioner Interaksi

No.	Jenis Interaksi	Modus Penilaian Pengujian
1	Gamepad	5
2	Leap Motion	4

Berdasarkan perolehan data tersebut, dapat diketahui bahwa penggunaan interaksi *gamepad* dalam lingkungan virtual memiliki nilai modus 5. Berdasarkan penilaian tersebut, penguji setuju dengan penggunaan interaksi dengan *gamepad.* Sedangkan interaksi Leap Motion, nilai modus yang diperoleh yaitu 4. Berdasarkan penilaian tersebut, penguji cukup setuju dengan penggunaan Leap Motion sebagai interaksi dalam lingkungan virtual.

Meskipun begitu, ada beberapa saran yang dilontarkan oleh para penguji terkait pengujian interaksi yang dilakukan. Penguji merasa lingkungan pengujian perlu dibenahi, terutama dengan lingkugan rumah. Selain itu, penguji juga mengeluhkan kurang nyamannya penggunaan Google Cardboard ketika melakukan pengujian.

LAMPIRAN A. LEMBAR PENGUJIAN



LEMBAR PENGUJIAN

"GESTUR TANGAN SEBAGAI INTERAKSI DAN KONTROL DALAM REALITAS VIRTUAL MENGGUNAKAN GOOGLE CARDBOARD DAN LEAP MOTION"

Nama : Diagnosa F.
Umur : 21
Pekerjaan : Mahasiswa

Karakteristik Pengguna

Silahkan lingkari (O) pada pilihan jawaban yang sesuai

1. Saya pernah menggunakan perangkat Realitas Virtual sebelumnya (Google Cardboard, Oculus Rift)
   (a) Pernah b. Tidak pernah
2. Saya sudah terbiasa menggunakan *gamepad* ketika memainkan *game*
   a. Ya (b) Tidak

Pengujian Interaksi

Jenis Interaksi	Waktu penyelesaian (detik)		
	Percobaan 1	Percobaan 2	Percobaan 3
Gamepad	02 : 12 : 6	01 : 48 : 1	01 : 12 : 2
Leap Motion	03 : 56 : 2	04 : 05 : 9	03 : 18 : 4

Kuesioner Pengujian

Silahkan isi centang (✓) di kolom sesuai dengan preferensi Anda

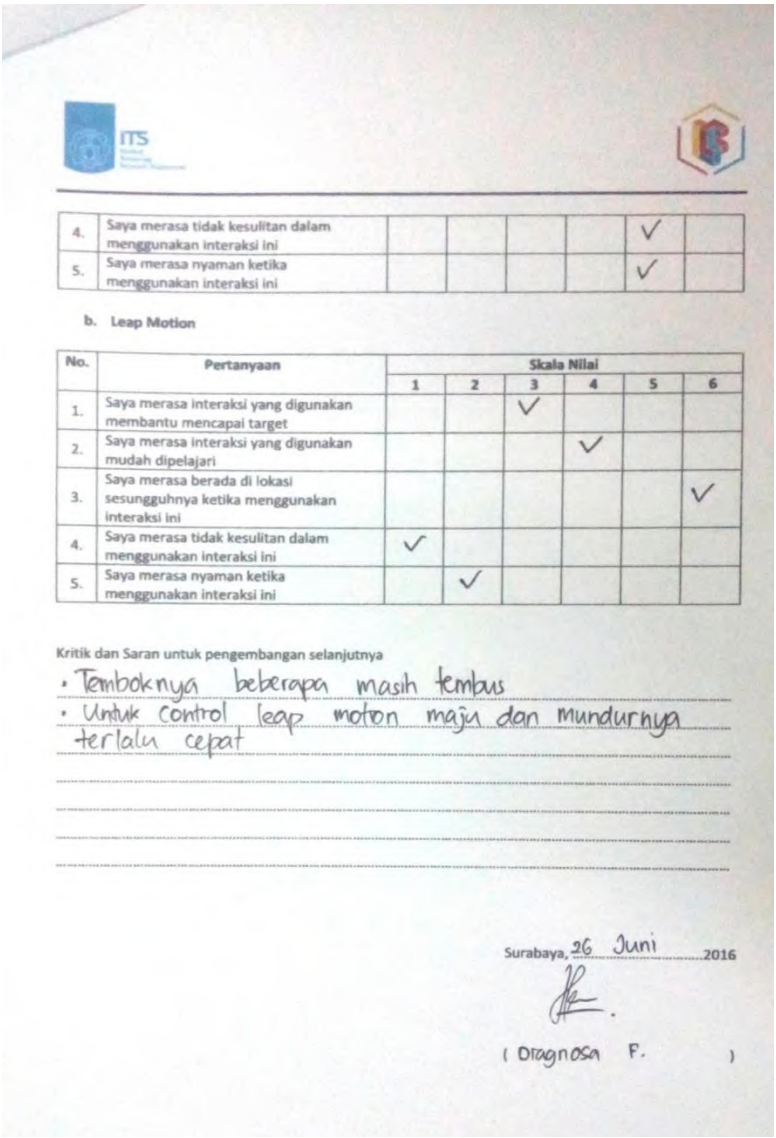
Skala Nilai :

1: Sangat Tidak Setuju 3: Kurang Setuju 5: Setuju
2: Tidak Setuju 4: Cukup Setuju 6: Sangat Setuju

a. ***Gamepad***

No.	Pertanyaan	Skala Nilai					
		1	2	3	4	5	6
1.	Saya merasa interaksi yang digunakan membantu mencapai target					✓	
2.	Saya merasa interaksi yang digunakan mudah dipelajari					✓	
3.	Saya merasa berada di lokasi sesungguhnya ketika menggunakan interaksi ini						✓

Gambar A.0.1 Lembar Pengujian Responden Pertama Bagian 1



4.	Saya merasa tidak kesulitan dalam menggunakan interaksi ini					✓	
5.	Saya merasa nyaman ketika menggunakan interaksi ini					✓	

b. **Leap Motion**

No.	Pertanyaan	Skala Nilai					
		1	2	3	4	5	6
1.	Saya merasa interaksi yang digunakan membantu mencapai target			✓			
2.	Saya merasa interaksi yang digunakan mudah dipelajari				✓		
3.	Saya merasa berada di lokasi sesungguhnya ketika menggunakan interaksi ini						✓
4.	Saya merasa tidak kesulitan dalam menggunakan interaksi ini	✓					
5.	Saya merasa nyaman ketika menggunakan interaksi ini		✓				

Kritik dan Saran untuk pengembangan selanjutnya

- Tombolnya beberapa masih tembus
- Untuk control leap motion maju dan mundurnya terlalu cepat

Surabaya, 26 Juni 2016

(Dragnosa F.)

Gambar A.0.2 Lembar Pengujian Responden Pertama Bagian 2

LEMBAR PENGUJIAN

"GESTUR TANGAN SEBAGAI INTERAKSI DAN KONTROL DALAM REALITAS VIRTUAL MENGGUNAKAN GOOGLE CARDBOARD DAN LEAP MOTION"

Nama : Ignatius Abraham S.
Umur : 22
Pekerjaan : Mahasiswa

Karakteristik Pengguna

Silahkan lingkari (O) pada pilihan jawaban yang sesuai

1. Saya pernah menggunakan perangkat Realitas Virtual sebelumnya (Google Cardboard, Oculus Rift)
   (a) Pernah b. Tidak pernah
2. Saya sudah terbiasa menggunakan *gamepad* ketika memainkan *game*
   (a) Ya b. Tidak

Pengujian Interaksi

Jenis Interaksi	Waktu penyelesaian (detik)		
	Percobaan 1	Percobaan 2	Percobaan 3
Gamepad	05 : 43 : 2	01 : 12 : 2	00 : 57 : 0
Leap Motion	01 : 59 : 7	02 : 04 : 0	01 : 28 : 9

Kuesioner Pengujian

Silahkan isi centang (✓) di kolom sesuai dengan preferensi Anda

Skala Nilai :

1: Sangat Tidak Setuju
2: Tidak Setuju
3: Kurang Setuju
4: Cukup Setuju
5: Setuju
6: Sangat Setuju

a. ***Gamepad***

No.	Pertanyaan	Skala Nilai					
		1	2	3	4	5	6
1.	Saya merasa interaksi yang digunakan membantu mencapai target					✓	
2.	Saya merasa interaksi yang digunakan mudah dipelajari						✓
3.	Saya merasa berada di lokasi sesungguhnya ketika menggunakan interaksi ini					✓	

Gambar A.0.3 Lembar Pengujian Responden Kedua Bagian 1

4.	Saya merasa tidak kesulitan dalam menggunakan interaksi ini				·	✓	
5.	Saya merasa nyaman ketika menggunakan interaksi ini				✓		

b. **Leap Motion**

No.	Pertanyaan	Skala Nilai					
		1	2	3	4	5	6
1.	Saya merasa interaksi yang digunakan membantu mencapai target					✓	
2.	Saya merasa interaksi yang digunakan mudah dipelajari					✓	
3.	Saya merasa berada di lokasi sesungguhnya ketika menggunakan interaksi ini					✓	
4.	Saya merasa tidak kesulitan dalam menggunakan interaksi ini				✓		
5.	Saya merasa nyaman ketika menggunakan interaksi ini				✓		

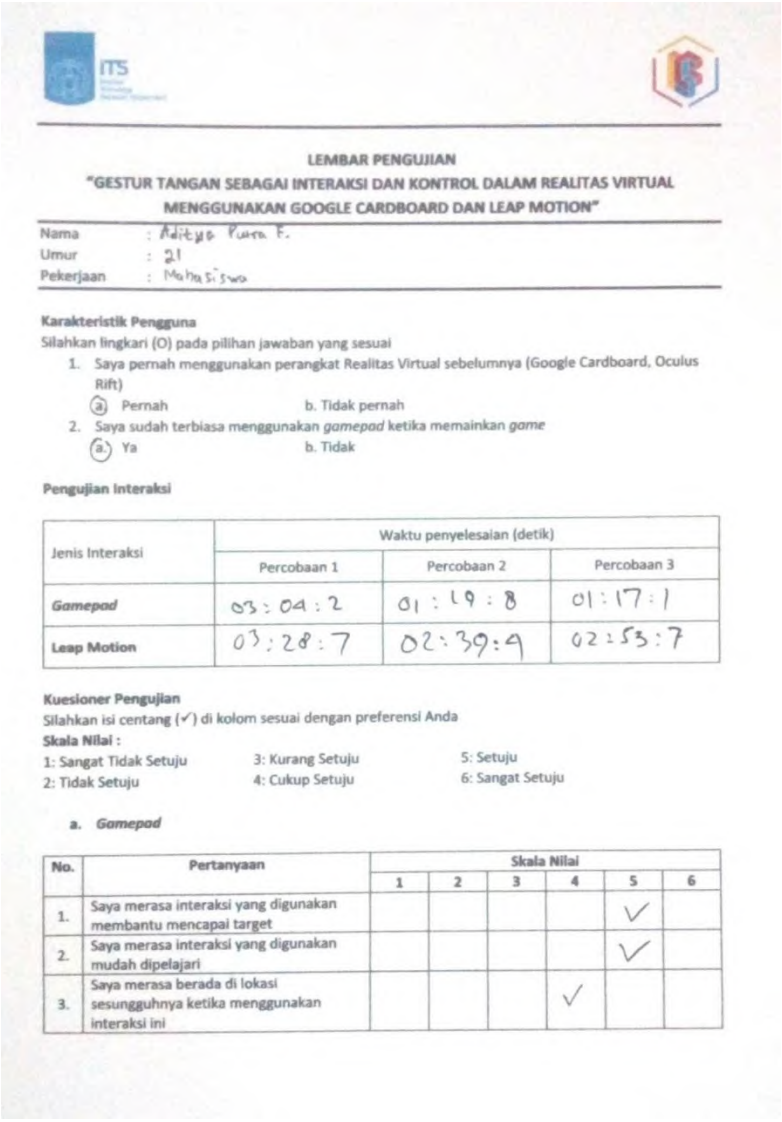
Kritik dan Saran untuk pengembangan selanjutnya

Untuk tempat pengujian adalah tempat yang bebas gangguan

Surabaya, 26 Juni 2016

(Ignatius Abraham S.)

Gambar A.0.4 Lembar Pengujian Responden Kedua Bagian 2



LEMBAR PENGUJIAN

"GESTUR TANGAN SEBAGAI INTERAKSI DAN KONTROL DALAM REALITAS VIRTUAL MENGGUNAKAN GOOGLE CARDBOARD DAN LEAP MOTION"

Nama : Aditya Putra F.
Umur : 21
Pekerjaan : Mahasiswa

Karakteristik Pengguna

Silahkan lingkari (O) pada pilihan jawaban yang sesuai

1. Saya pernah menggunakan perangkat Realitas Virtual sebelumnya (Google Cardboard, Oculus Rift)
   (a) Pernah b. Tidak pernah
2. Saya sudah terbiasa menggunakan *gamepad* ketika memainkan *game*
   (a.) Ya b. Tidak

Pengujian Interaksi

Jenis Interaksi	Waktu penyelesaian (detik)		
	Percobaan 1	Percobaan 2	Percobaan 3
Gamepad	03 : 04 : 2	01 : 19 : 8	01 : 17 : 1
Leap Motion	03 : 28 : 7	02 : 39 : 9	02 : 53 : 7

Kuesioner Pengujian

Silahkan isi centang (✓) di kolom sesuai dengan preferensi Anda

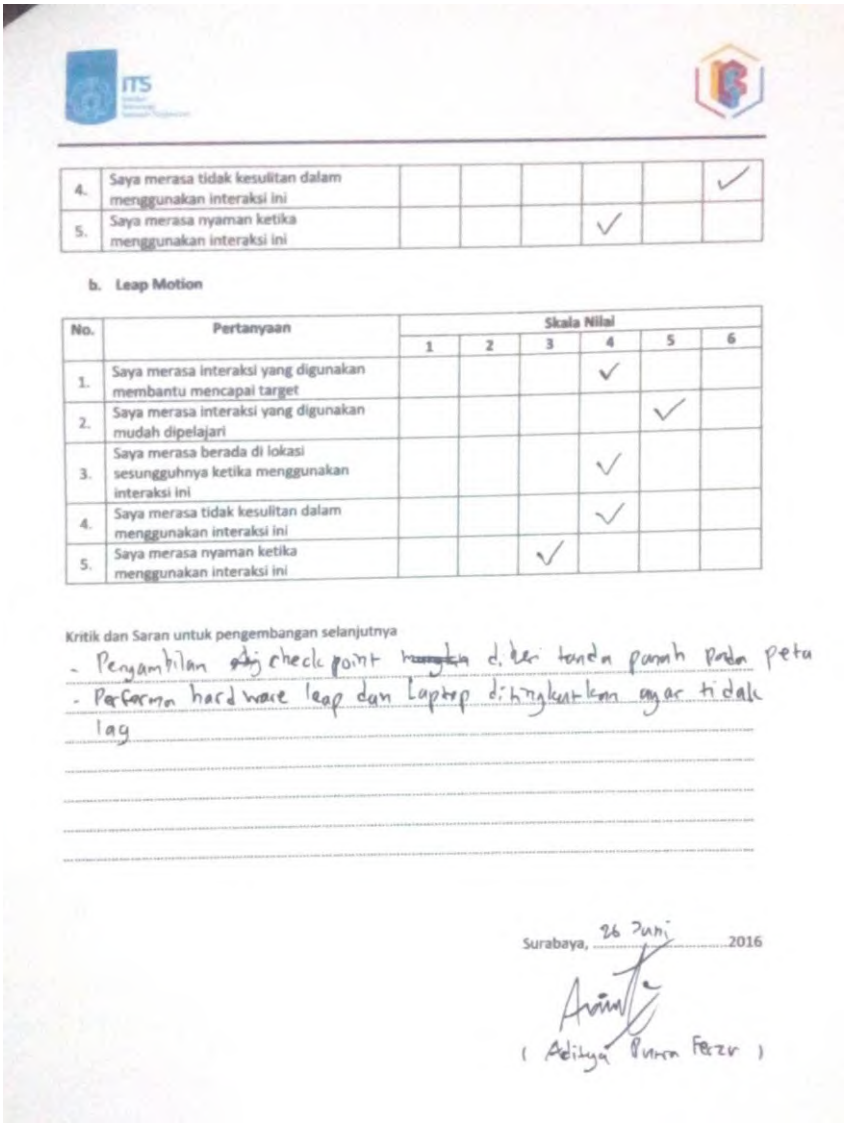
Skala Nilai :

1: Sangat Tidak Setuju 3: Kurang Setuju 5: Setuju
2: Tidak Setuju 4: Cukup Setuju 6: Sangat Setuju

a. ***Gamepad***

No.	Pertanyaan	Skala Nilai					
		1	2	3	4	5	6
1.	Saya merasa interaksi yang digunakan membantu mencapai target					✓	
2.	Saya merasa interaksi yang digunakan mudah dipelajari					✓	
3.	Saya merasa berada di lokasi sesungguhnya ketika menggunakan interaksi ini				✓		

Gambar A.0.5 Lembar Pengujian Responden Ketiga Bagian 1



4.	Saya merasa tidak kesulitan dalam menggunakan interaksi ini						✓
5.	Saya merasa nyaman ketika menggunakan interaksi ini				✓		

b. **Leap Motion**

No.	Pertanyaan	Skala Nilai					
		1	2	3	4	5	6
1.	Saya merasa interaksi yang digunakan membantu mencapai target				✓		
2.	Saya merasa interaksi yang digunakan mudah dipelajari					✓	
3.	Saya merasa berada di lokasi sesungguhnya ketika menggunakan interaksi ini				✓		
4.	Saya merasa tidak kesulitan dalam menggunakan interaksi ini				✓		
5.	Saya merasa nyaman ketika menggunakan interaksi ini			✓			

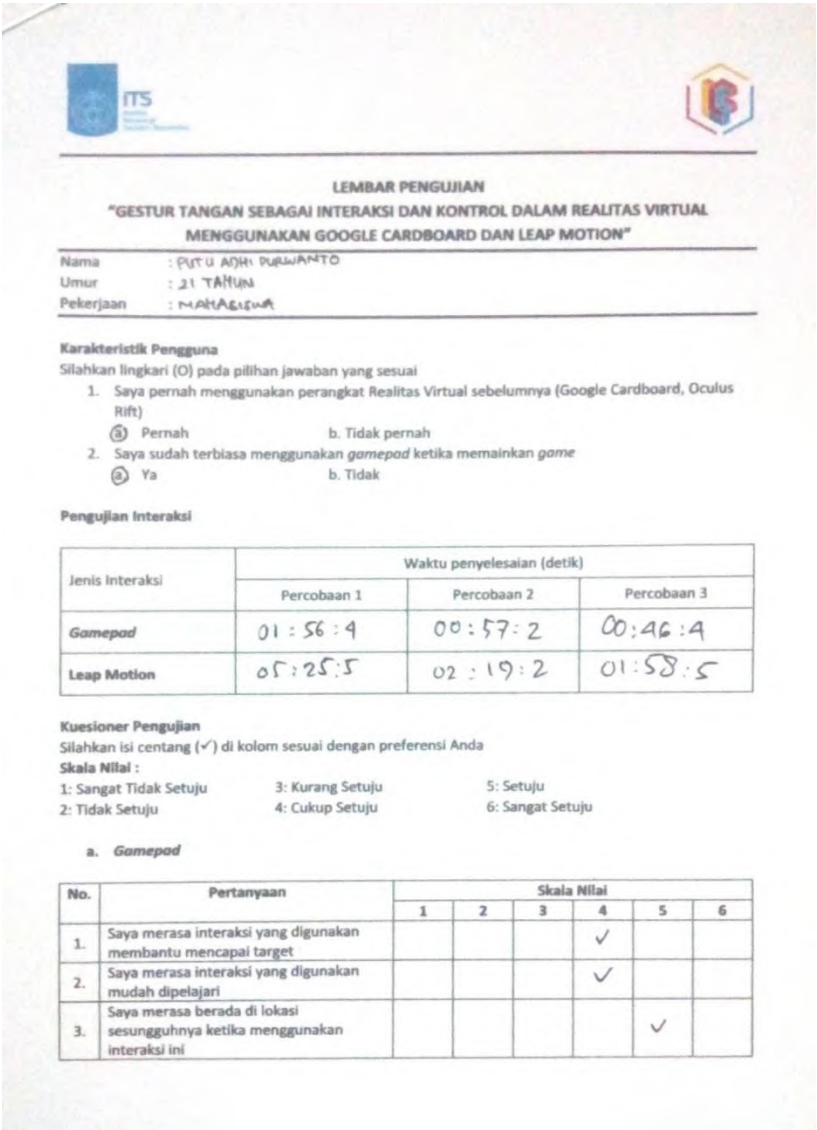
Kritik dan Saran untuk pengembangan selanjutnya

- Pengambilan ~~obj~~ checkpoint ~~mungkin~~ diberi tanda pasti pada peta
- Performa hardware leap dan laptop ditingkatkan agar tidak lag

Surabaya, 26 Juni 2016

(Aditya Putra Fezer)

Gambar A.0.6 Lembar Pengujian Responden Ketiga Bagian 2



LEMBAR PENGUJIAN

"GESTUR TANGAN SEBAGAI INTERAKSI DAN KONTROL DALAM REALITAS VIRTUAL MENGGUNAKAN GOOGLE CARDBOARD DAN LEAP MOTION"

Nama : PUTU ADHI PURWANTO
Umur : 21 TAHUN
Pekerjaan : MAHASISWA

Karakteristik Pengguna

Silahkan lingkari (O) pada pilihan jawaban yang sesuai

1. Saya pernah menggunakan perangkat Realitas Virtual sebelumnya (Google Cardboard, Oculus Rift)
   a. Pernah b. Tidak pernah
2. Saya sudah terbiasa menggunakan *gamepad* ketika memainkan *game*
   a. Ya b. Tidak

Pengujian Interaksi

Jenis Interaksi	Waktu penyelesaian (detik)		
	Percobaan 1	Percobaan 2	Percobaan 3
Gamepad	01 : 56 : 9	00 : 57 : 2	00 : 46 : 9
Leap Motion	05 : 25 : 5	02 : 19 : 2	01 : 58 : 5

Kuesioner Pengujian

Silahkan isi centang (✓) di kolom sesuai dengan preferensi Anda

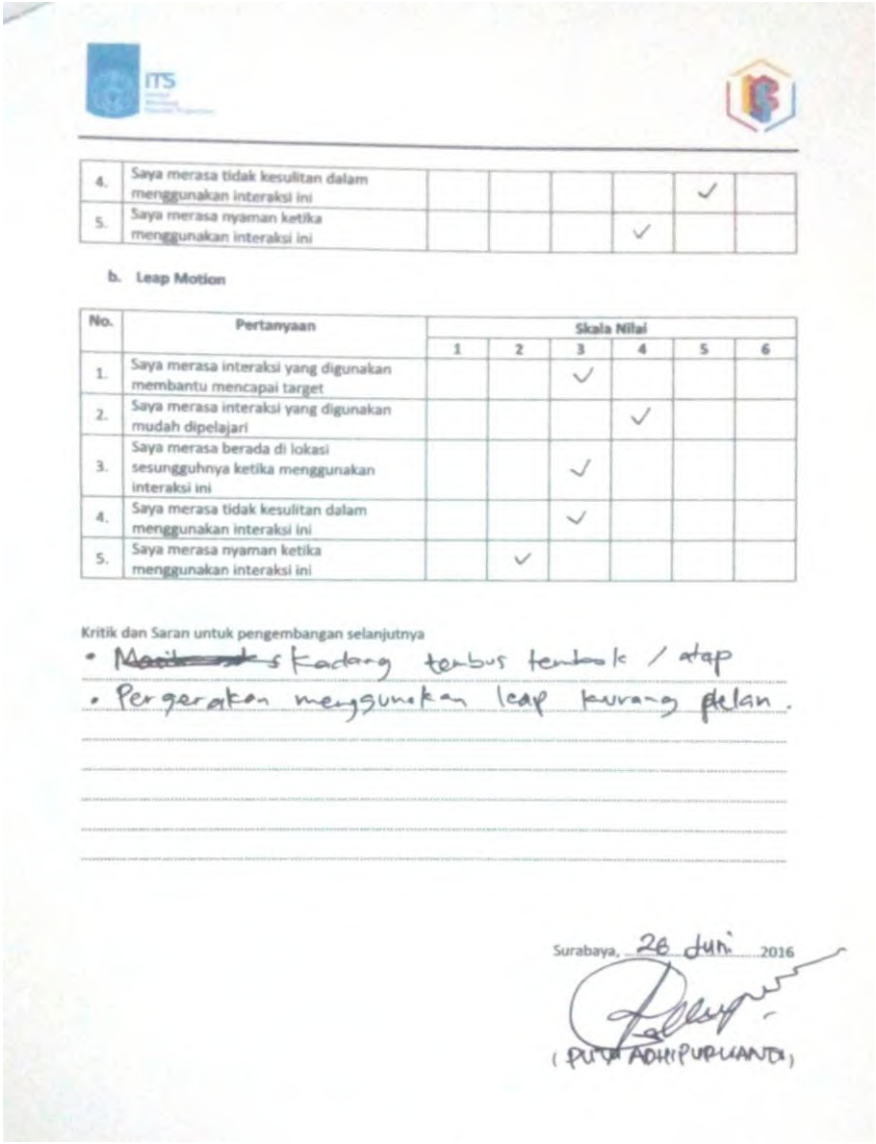
Skala Nilai :

1: Sangat Tidak Setuju
2: Tidak Setuju
3: Kurang Setuju
4: Cukup Setuju
5: Setuju
6: Sangat Setuju

a. ***Gamepad***

No.	Pertanyaan	Skala Nilai					
		1	2	3	4	5	6
1.	Saya merasa interaksi yang digunakan membantu mencapai target				✓		
2.	Saya merasa interaksi yang digunakan mudah dipelajari				✓		
3.	Saya merasa berada di lokasi sesungguhnya ketika menggunakan interaksi ini					✓	

Gambar A.0.7 Lembar Pengujian Responden Keempat Bagian 1



4.	Saya merasa tidak kesulitan dalam menggunakan interaksi ini					✓	
5.	Saya merasa nyaman ketika menggunakan interaksi ini				✓		

b. **Leap Motion**

No.	Pertanyaan	Skala Nilai					
		1	2	3	4	5	6
1.	Saya merasa interaksi yang digunakan membantu mencapai target			✓			
2.	Saya merasa interaksi yang digunakan mudah dipelajari				✓		
3.	Saya merasa berada di lokasi sesungguhnya ketika menggunakan interaksi ini			✓			
4.	Saya merasa tidak kesulitan dalam menggunakan interaksi ini			✓			
5.	Saya merasa nyaman ketika menggunakan interaksi ini		✓				

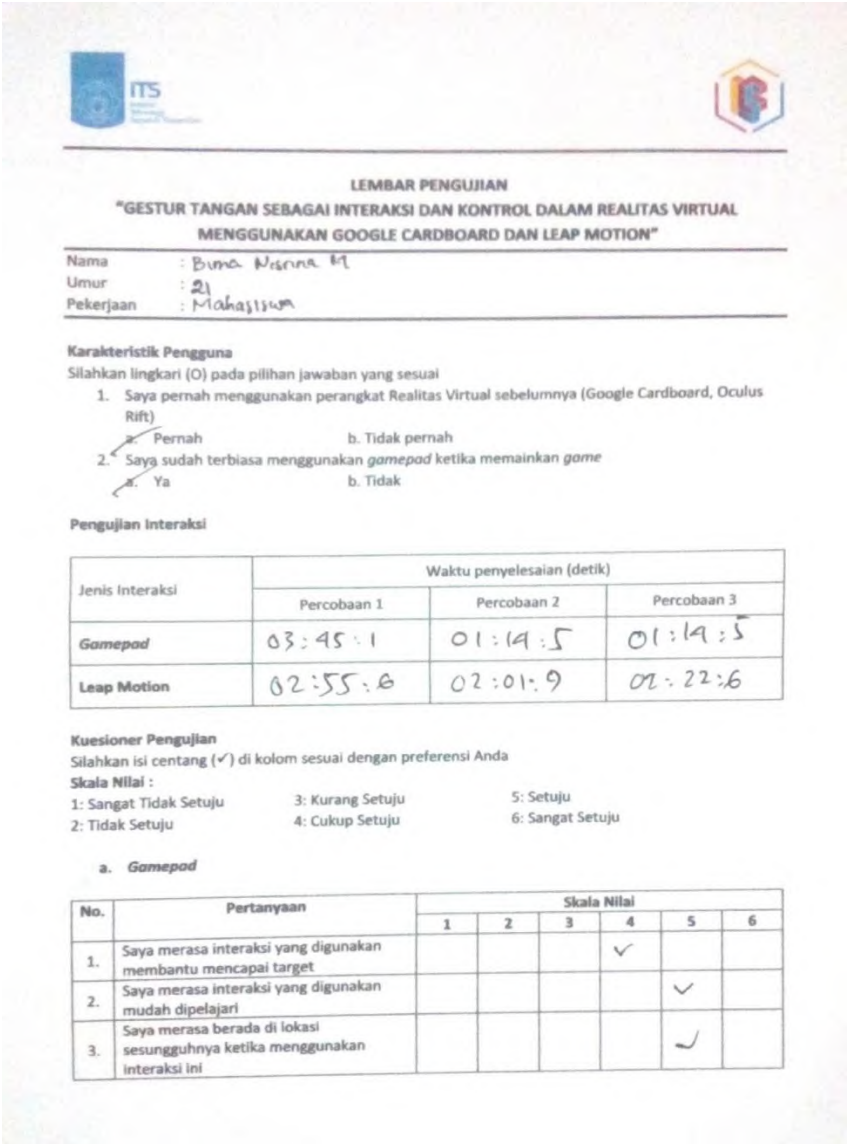
Kritik dan Saran untuk pengembangan selanjutnya

- [redacted] kadang terbus tembok / atap
- Pergerakan menggunakan leap kurang jelas.

Surabaya, 28 juni 2016

(PUTU ADHI PURNANDI)

Gambar A.0.8 Lembar Pengujian Responden Keempat Bagian 2



LEMBAR PENGUJIAN

"GESTUR TANGAN SEBAGAI INTERAKSI DAN KONTROL DALAM REALITAS VIRTUAL MENGGUNAKAN GOOGLE CARDBOARD DAN LEAP MOTION"

Nama : Bima Nesma M
Umur : 21
Pekerjaan : Mahasiswa

Karakteristik Pengguna

Silahkan lingkari (O) pada pilihan jawaban yang sesuai

1. Saya pernah menggunakan perangkat Realitas Virtual sebelumnya (Google Cardboard, Oculus Rift)
   a. Pernah b. Tidak pernah
2. Saya sudah terbiasa menggunakan *gamepad* ketika memainkan *game*
   a. Ya b. Tidak

Pengujian Interaksi

Jenis Interaksi	Waktu penyelesaian (detik)		
	Percobaan 1	Percobaan 2	Percobaan 3
Gamepad	03:45:1	01:19:5	01:19:5
Leap Motion	02:55:6	02:01:9	02:22:6

Kuesioner Pengujian

Silahkan isi centang (✓) di kolom sesuai dengan preferensi Anda

Skala Nilai :

1: Sangat Tidak Setuju
2: Tidak Setuju
3: Kurang Setuju
4: Cukup Setuju
5: Setuju
6: Sangat Setuju

a. ***Gamepad***

No.	Pertanyaan	Skala Nilai					
		1	2	3	4	5	6
1.	Saya merasa interaksi yang digunakan membantu mencapai target				✓		
2.	Saya merasa interaksi yang digunakan mudah dipelajari					✓	
3.	Saya merasa berada di lokasi sesungguhnya ketika menggunakan interaksi ini					✓	

Gambar A.0.9 Lembar Pengujian Responden Kelima Bagian 1

4.	Saya merasa tidak kesulitan dalam menggunakan interaksi ini					V	
5.	Saya merasa nyaman ketika menggunakan interaksi ini					V	

b. **Leap Motion**

No.	Pertanyaan	Skala Nilai					
		1	2	3	4	5	6
1.	Saya merasa interaksi yang digunakan membantu mencapai target				V		
2.	Saya merasa interaksi yang digunakan mudah dipelajari				V		
3.	Saya merasa berada di lokasi sesungguhnya ketika menggunakan interaksi ini					V	
4.	Saya merasa tidak kesulitan dalam menggunakan interaksi ini			V			
5.	Saya merasa nyaman ketika menggunakan interaksi ini			V			

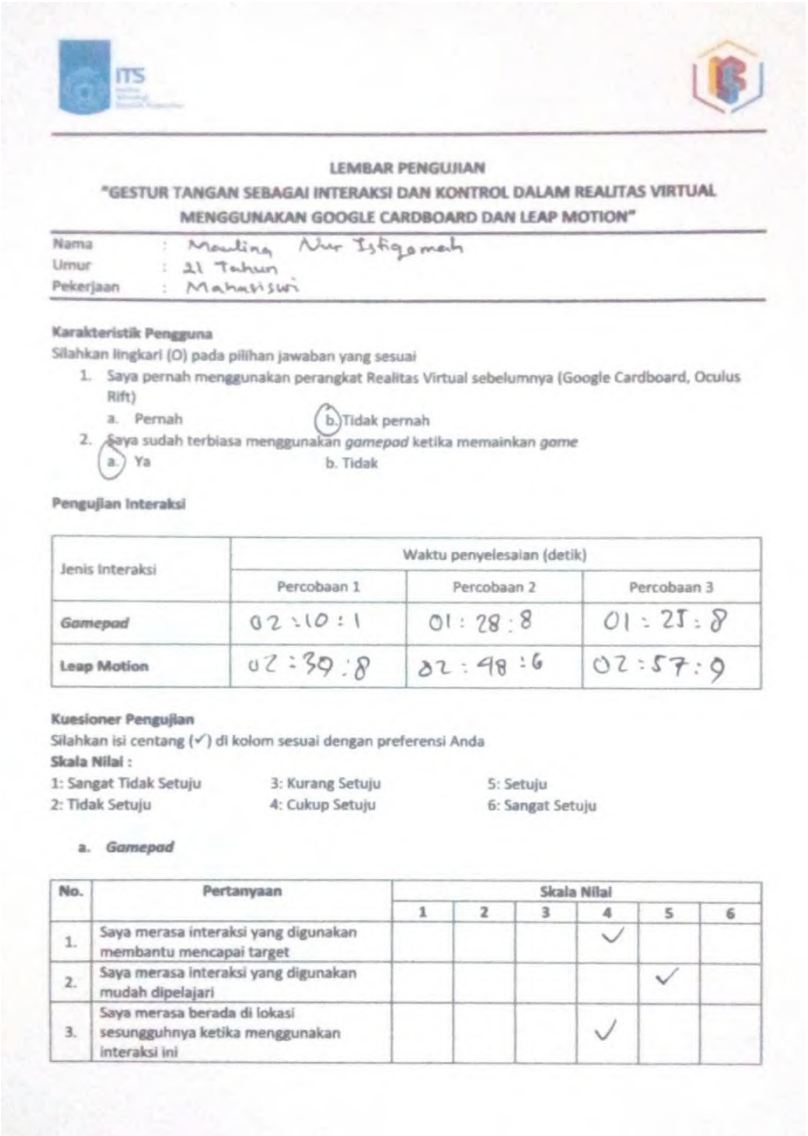
Kritik dan Saran untuk pengembangan selanjutnya

Gestur belok dan mundurnya kurang mudah digunakan

Surabaya, 26 Juni 2016

(Nanng)

Gambar A.0.10 Lembar Pengujian Responden Kelima Bagian 2



LEMBAR PENGUJIAN

"GESTUR TANGAN SEBAGAI INTERAKSI DAN KONTROL DALAM REALITAS VIRTUAL MENGGUNAKAN GOOGLE CARDBOARD DAN LEAP MOTION"

Nama : Moulina Nur Istiqomah
Umur : 21 Tahun
Pekerjaan : Mahasiswi

Karakteristik Pengguna

Silahkan lingkari (O) pada pilihan jawaban yang sesuai

1. Saya pernah menggunakan perangkat Realitas Virtual sebelumnya (Google Cardboard, Oculus Rift)
   a. Pernah b. Tidak pernah
2. Saya sudah terbiasa menggunakan *gamepad* ketika memainkan *game*
   a. Ya b. Tidak

Pengujian Interaksi

Jenis Interaksi	Waktu penyelesaian (detik)		
	Percobaan 1	Percobaan 2	Percobaan 3
Gamepad	02:10:1	01:28:8	01:25:8
Leap Motion	02:39:8	02:48:6	02:57:9

Kuesioner Pengujian

Silahkan isi centang (✓) di kolom sesuai dengan preferensi Anda

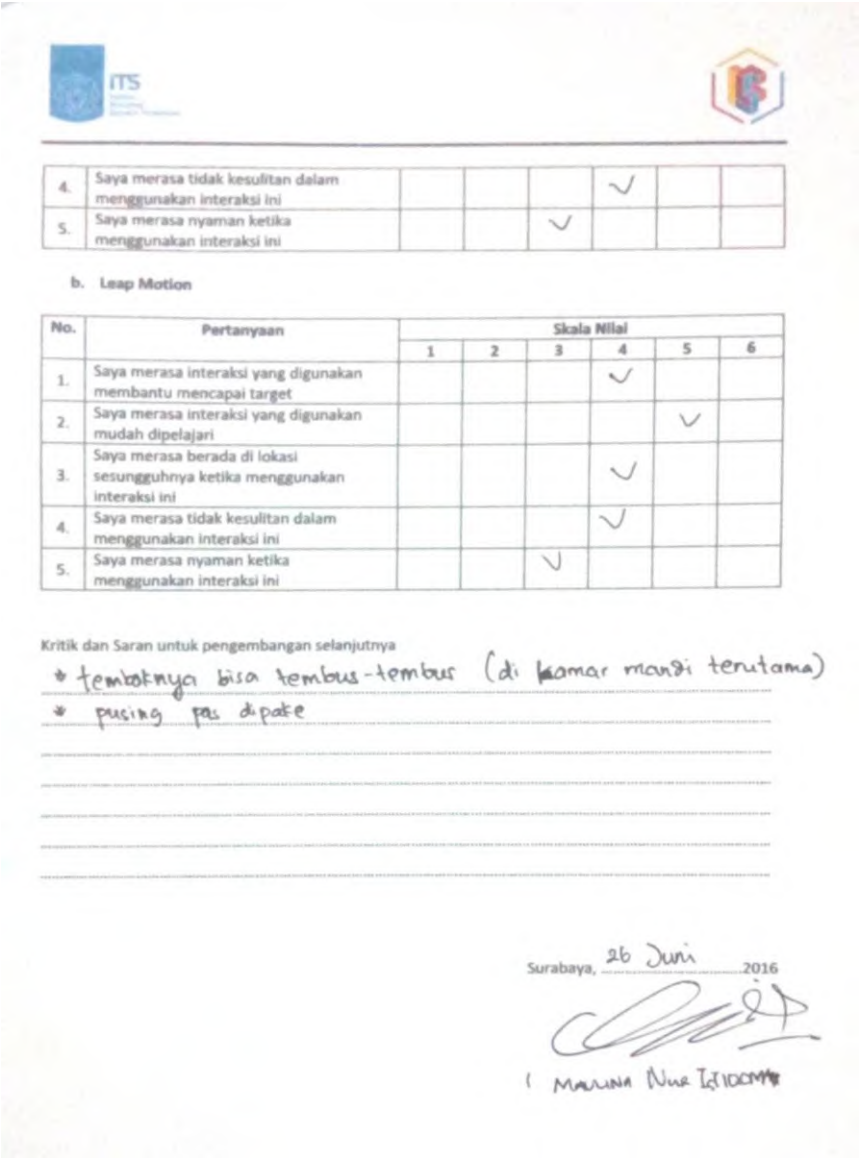
Skala Nilai :

1: Sangat Tidak Setuju
2: Tidak Setuju
3: Kurang Setuju
4: Cukup Setuju
5: Setuju
6: Sangat Setuju

a. ***Gamepad***

No.	Pertanyaan	Skala Nilai					
		1	2	3	4	5	6
1.	Saya merasa interaksi yang digunakan membantu mencapai target				✓		
2.	Saya merasa interaksi yang digunakan mudah dipelajari					✓	
3.	Saya merasa berada di lokasi sesungguhnya ketika menggunakan interaksi ini				✓		

Gambar A.0.11 Lembar Pengujian Responden Keenam Bagian 1



4.	Saya merasa tidak kesulitan dalam menggunakan interaksi ini				✓		
5.	Saya merasa nyaman ketika menggunakan interaksi ini			✓			

b. **Leap Motion**

No.	Pertanyaan	Skala Nilai					
		1	2	3	4	5	6
1.	Saya merasa interaksi yang digunakan membantu mencapai target				✓		
2.	Saya merasa interaksi yang digunakan mudah dipelajari					✓	
3.	Saya merasa berada di lokasi sesungguhnya ketika menggunakan interaksi ini				✓		
4.	Saya merasa tidak kesulitan dalam menggunakan interaksi ini				✓		
5.	Saya merasa nyaman ketika menggunakan interaksi ini			✓			

Kritik dan Saran untuk pengembangan selanjutnya

* tombolnya bisa tembus-tembus (di kamar mandi terutama)
* pusing pas dipake

Surabaya, 26 Juni 2016

(Marlina Nur Istiqomah)

Gambar A.0.12 Lembar Pengujian Responden Keenam Bagian 2

BAB VI
KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diambil selama pengerjaan Tugas Akhir serta saran-saran tentang pengembangan yang dapat dilakukan terhadap Tugas Akhir ini di masa yang akan datang.

6.1. Kesimpulan

Dari hasil selama proses perancangan, implementasi, serta pengujian dapat diambil kesimpulan sebagai berikut:

1. Dalam mendemonstrasikan teknologi realitas virtual dengan interaksi tangan melalui Leap Motion, dibuat sebuah skenario pengambilan beberapa *checkpoint* yang lokasinya ditentukan pada sebuah model rumah 3D. Selama pengguna mengambil beberapa *checkpoint*, pencatatan waktu dilakukan sehingga waktu tempuh pengguna dapat diketahui setelah pengguna berhasil mengambil semua *checkpoint*.
2. Gestur tangan untuk menggerakkan karakter pada lingkungan virtual memanfaatkan sumbu x dan z dalam koordinat Unity. Sumbu x berguna untuk memanipulasi gerakan horizontal yaitu kiri dan kanan. Gestur tangan yang sesuai dengan sumbu x adalah mencondongkan tangan secara horizontal. Sumbu z berguna untuk memanipulasi gerakan vertikal yaitu maju dan mundur. Gestur tangan yang sesuai dengan sumbu z adalah arah telapak tangan ke depan dan belakang. Selain itu, untuk mengetahui keadaan bergerak dan berhenti, aplikasi meninjau keadaan tangan ketika membuka dan mengepal.
3. Ditinjau dari hasil pengujian interaksi, dapat disimpulkan bahwa interaksi dengan menggunakan *gamepad* lebih mudah diadaptasi oleh pengguna dibandingkan dengan menggunakan Leap Motion.

4. Hasil kuesioner interaksi menunjukkan bahwa pengguna lebih setuju penggunaan interaksi dengan menggunakan *gamepad* dibandingkan dengan Leap Motion dalam lingkungan realitas virtual.

6.2. Saran

Berikut saran-saran untuk pengembangan dan perbaikan sistem di masa yang akan datang. Diantaranya adalah sebagai berikut:

1. Meningkatkan performa perangkat keras yang digunakan untuk menjalankan aplikasi yang memanfaatkan interaksi melalui Leap Motion.
2. Adanya dukungan Leap Motion API pada *smartphone* secara *native* sehingga memudahkan pengembangan interaksi Leap Motion pada *smartphone*.
3. Sebaiknya responden yang melakukan pengujian belum pernah menggunakan *gamepad* dan Leap Motion.

DAFTAR PUSTAKA

[1] D. Bowman, D. Koller and L. F. Hodges, "A methodology for the evaluation of travel techniques for immersive virtual environments," *Virtual Reality,* vol. 3, no. 2, pp. 120 - 131, 1998.

[2] T. Mazuryk and G. Michael, Virtual Reality : History, Applications, Technology and Future, Austria: Institute of Computer Graphics, Vienna University of Technology, 1996.

[3] J. M. Oscilada, "Virtual Reality Times," [Online]. Available: http://www.virtualrealitytimes.com/2015/05/24/chart-fov-field-of-view-vr-headsets/. [Accessed 2015].

[4] G. Developers, "Google VR SDK for Unity," Alphabet Inc., [Online]. Available: https://developers.google.com/vr/unity/. [Accessed 11 June 2016].

[5] T. VR, "Trinus VR : How It Works," [Online]. Available: http://trinusvr.com/help/how-it-works/. [Accessed June 2016].

[6] V. Llopis, "Introducing the Skeletal Tracking Model," The Motion Report, 6 May 2013. [Online]. Available: http://themotionreport.com/evolution-of-leap-motion-controller/. [Accessed 25 May 2016].

[7] A. Colgan, "Introducing the Skeletal Tracking Model," Leap Motion, Inc, 9 August 2014. [Online]. Available: http://blog.leapmotion.com/hardware-to-software-how-does-the-leap-motion-controller-work/. [Accessed 25 May 2016].

[8] L. Motion, "Unity Plugin Overview," Leap Motion Inc., [Online]. Available:

https://developer.leapmotion.com/documentation/v2/unity/unity/Unity_Overview.html. [Accessed June 2016].

[9] L. Motion, "HandModel - Leap Motion Unity Documentation," Leap Motion Inc., [Online]. Available: https://developer.leapmotion.com/documentation/v2/unity/unity/Unity.HandModel.html. [Accessed June 2016].

[10] L. Motion, "Hand - Leap Motion Unity Documentation," Leap Motion Inc., [Online]. Available: https://developer.leapmotion.com/documentation/v2/unity/api/Leap.Hand.html. [Accessed June 2016].

[11] Unity, "Game engine, tools and multi platfrom," Unity, [Online]. Available: http://unity3d.com/unity. [Accessed 9 april 2014].

[12] Y. Sato, M. Saito and K. Hideki, "Real-time input of 3D pose and gestures of a user's hand and its applications for HCI," *Virtual Reality 2001,* pp. 79-86, 2001.

[13] C. Khundam, "First Person Movement Control with Palm Normal," *12th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, pp. 325-326, 2015.

BIODATA PENULIS

Tri Sutrisno Nusantara, lahir pada tanggal 16 April 1995 di Surabaya. Saat ini sedang menempuh perguruan tinggi di Institut Teknologi Sepuluh November jurusan Teknik Informatika Fakultas Teknologi Informasi pada tahun 2012. Terlibat aktif pada organisasi mahasiswa tingkat jurusan maupun fakultas, antara lain Departemen Media Informasi Himpunan Mahasiswa Teknik Computer-Informatika (HMTC) ITS sebagai staff tahun 2013 – 2014 dan sebagai staff ahli pada tahun 2014 – 2015, Departemen Information Media BEM Fakultas Teknologi Informasi (FTIf) sebagai staff tahun 2013 – 2014, dan sebagai kepala biro bidang media kreatif pada tahun 2014 – 2015.

Apabila ingin bertanya lebih lanjut, silahkan kirim *email* ke tri.sutrisno.n16@gmail.com.